ОПТИМИЗИРАНЕ СКОРОСТТА НА ХОДЕНЕ НА РОБОТ ЧРЕЗ ГЕНЕТИЧЕН АЛГОРИТЪМ

Ася Тоскова, Станимир Стоянов

Резюме: *Настоящата разработка представя резултати от реализиран генетичен алгоритъм за оптимизиране скоростта на движение на хуманоиден робот Нао в симулирана среда. Предложеният подход за* ***машинно*** *самообучение е фокусиран върху минимизиране на времената за завъртане на ставите на актуаторите, което осигурява повишаване на общата скорост на движение.*

Ключови думи: *хуманоиден крачещ робот Нао, генетичен алгоритъм, бързо ходене, симулирана среда, машинно обучение, изкуствен интелект*

Въведение

В Пловдивския университет е създадено пространство за виртуално образование (ВОП), което интегрира хетерогенни технологии и педагогически подходи (Stoyanov, 2016, Valkanov, Stoyanov, Valkanova, 2016) и представлява инфраструктура за предоставяне на електронни образователни услуги и учебно съдържание (Stoynov, Zedan, Doychev, Valkanova, Stoyanova-Doycheva, Valkanov, 2016, Doychev, Stoyanova-Doycheva, Stoyanov, Ivanova, 2016, Stoyanov, Stoyanova-Doycheva, Doychev, Gramatova, 2016). Виртуалната учебна среда постоянно се разширява като търси най-добрия подход за усъвършенстване на учебния процес, за постигане на по-голяма динамика, гъвкавост и адаптивност. В термините на интернет на нещата ВОП може да се дефинира като екологична система, в която всички участници са взаимосвързани и са способни интелигентно да коригират собственото си поведение, базирано на наблюдение, анализ, натрупване на знания, разсъждения и решения. Интелигентните агенти са едни от участниците в пространството. Те са специализирани за различни дейности – персоналните асистенти доставят образователни услуги и се адаптират спрямо желанията и навиците на потребителя, с когото контактуват; сървърните агенти извличат ресурси от дигиталните бази; оперативните агенти свързват сървърните агенти с персоналните асистенти и транспортират образователните ресурси и услуги; връзка с физическия свят и ВОП осъществяват гардовете. Всички те могат да работят както в софтуерна среда, така и имплементирани в различни хардуерни решения, в това число роботи и роботизирани системи.

Агентите, работещи върху роботи, могат да отговарят за различни задачи. Изключително актуална и съществена се явява задачата за мобилността на робота, като особен интерес представляват хуманоидните роботи и контролът на техните физически характеристики. Най-съществено предимство на такива роботи е възможността им за придвижване, подобно на човека. Известно е, че ходенето на два крака е най-стабилният вариант за преминаване през пресечена местност. Затова ходещите роботи могат да заменят човека в изпълнението на опасни за него задачи в труднопроходима среда. Оттук произлиза и основната идея за разработване на модул за самообучение на агент, опериращ върху хуманоиден робот (Стоянов, Тоскова, Тодоров, Русев, 2017). Реализирането на тази идея се свежда до решаването на много отделни задачи, като една от тях е оптимизиране скоростта на движение на робота чрез самообучение. В тази статия са представени резултатите от изпълнението на тази задача.

Свързани изследвания

Усъвършенстването на движението на хуманоиден робот може да се осъществи чрез оптимизация на кинематичните и динамичните му характеристики. За целта особено подходящ е генетичният алгоритъм (ГА). Той е вероятностен, итеративен метод за търсене и оптимизация, който може да реши мултимодални проблеми с NP-complete-сложност в условията на ресурсно-времеви ограничения (Наков, Добриков, 2015). ГА е вдъхновен от теорията на Дарвин за биологичната еволюция и намира най-доброто решение в дадено пространство на търсене (Holland, 1986). Доказано е, че ГА е ефективно средство за търсене в сложни пространства (Goldberg, 1989), при което намира добро решение, въпреки че няма гаранция, че то е оптимално.

Съществуват множество разработки за усъвършенстване на ходенето при хуманоидните роботи чрез генетични алгоритми. Някои подходи предлагат оптимизация на мултиагентното поведение при избор на действие, което води до повишаване на скоростта на движение (Girardi, Kooijman, Wiggers, Visser, 2013). Други използват концепцията за точката с нулев момент и обърнатото махало, за да осигурят стабилна траектория (Graf, Röfer, 2010, Zhao, Qi, Yan, Zhu, 2009). Има решения, при които се оптимизират синусоидалните колебания на центъра на масите, като генетичният алгоритъм намира оптималните коефициенти на осигуряващите обучение невронни мрежи (Liu, Wang, Goodman, Chen, 2016, Endo, Yamasaki, Maeno, Kitan, 2002). Постигнат е напредък за подобряване на стабилността на ходенето и чрез модели, използващи обратна връзка от движението на хора (Meriçli, Veloso, 2010).

Реализация

Подходът за реализация продробно е описан в (Toskova, 2017).

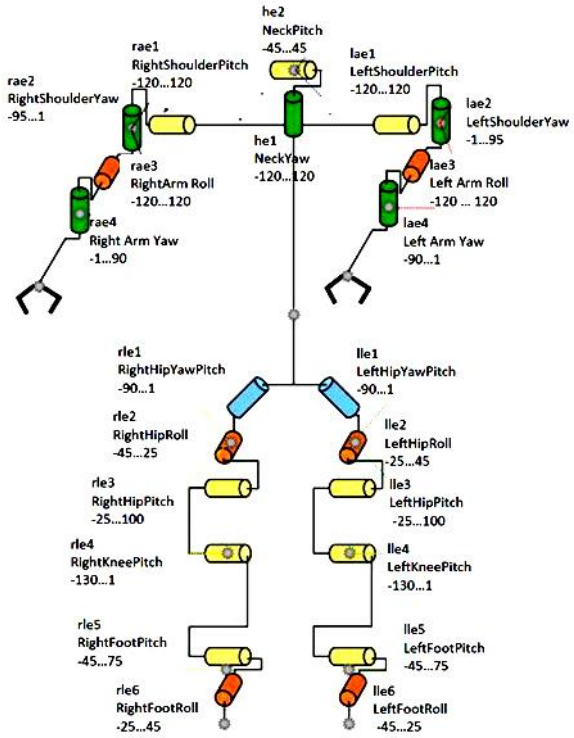
Платформа. За целите на изследването е използван роботът Nao на френската компания Aldebaran Robotics. Роботът и околната му среда са симулирани чрез официалния 3D симулатор SimSpark на световното RoboCup Soccer състезание в 3D Симулационната лига (RoboCup Soccer Server Maintenance Group, 2016). Комуникацията между агента и SimSpark сървъра се осъществява циклично на всеки 20 ms, посредством съобщения през TCP протокол.

Интелигентният агент, посредством който Nao действа, и алгоритъмът за обучение са реализирани на Java чрез RoboNewbie фреймуърк (Domanska, Burkhard, AI-Group, 2016), а движенията на робота, се задават с Motion Editor (AI-Group, 2013), като външен текстов файл. Всяко движение представлява поредица от ключови кадри (зададени скорости на завъртане на всяка става), които се описват и изпълняват просто, имат добър контрол върху детайлите и са подходящи за кратки обособени действия (крачка напред, крачка назад, ставане, ритане, въртене, ...).

В средата на SimSpark 3D кинематичните двойки на Nao са свързани с 22 активни цилиндрични стави, всяка с по една степен на свобода (Фигура 1).

Достигането на дадена поза се характеризира с преместване на кинематичните двойки на робота с определена скорост. При движението на Nao чрез ключови кадри са известни началното (x, y, z) и крайното положение (x', y', z'). Според принципите на обратната кинематика при пространствено-времевия модел, при известни начална и крайна поза е необходимо да се намери обратната матрица на ротация и транслация, която ни дава независимите параметри на движение – скоростта на транслация на звената на кинематичните вериги $V(t)$ и скоростта на завъртане на ставите $w(t)$ (Burkhard, 2016). Времето на завъртане на ставите и времето на транслация на звената е

едно и също (t). Подобряването на това време, при зададени предварително ъгли на завъртане на ставите, ще позволи определянето на оптималната скорост на движение.



Фигура 1.

Генетичен алгоритъм. Целта на генетичния алгоритъм е да намери евентуално най-доброто решение (индивид) в пространството на търсене (популация). Индивидът притежава хромозома. Хромозомата е вектор от гени, за подобряването на които се използват операторите селекция, кръстосване и мутация. Критерий за най-добро решение е фитнес функцията.

Един цикъл на ходене (действието „крачка напред“) съдържа няколко различни поредици от ключови кадри. Всеки ключов кадър включва време t и ъгли θ_i ($i \in [1, 22]$) на завъртане на ставите (таблица 1), като всяка става има една степен на свобода.

	Time	Joint 1	Joint 2	Joint 3	…	Joint 22
Pose J_i	t $[ms]$	θ_1 [°]	θ_2 [°]	θ_3 [°]		θ_{22} [°]

Таблица 1.

Всяка поза на робота се задава с вектора $J_i = [\omega_1, \omega_2, \dots \omega_n]$, където n е броят степени на свобода (в случая n = 22). Позите, през които роботът преминава, за да осъществи един цикъл на ходене, са k-броя: $J_1, J_2, \dots J_k$. Матрицата J (k x n) представлява нашият индивид, а хромозомата му е едномерен вектор с дължина k и с алели – стойностите на времето t.

Кодирането на гените е по стойност, тъй като времето t може да заема всички цели положителни числа, кратни на 20 (1 цикъл на комуникация е 20 ms). Първоначално гените се генерират на случаен принцип.

Стойностите на ъглите на завъртане на ставите θ са зададени от текстов файл, който съдържа актуалната поредица от ключови кадри. Те се движат в различен диапазон, като максималните граници са $[-120.0°, +120.0°]$ (виж фигура 1). На този етап стойностите на ъглите не се променят по време на обучението.

Както се вижда, ставите могат да се въртят по и обратно на часовниковата стрелка, като скоростта им е в диапазона $\approx [-2\pi, +2\pi]$.

В началото на алгоритъма създаваме начална популация от N индивида с произволно генерирани гени (времена). Големината на популацията определя пространството на търсене и съответно бързината на алгоритъма и качеството на намереното решение. Нашият алгоритъм работи почти еднакво добре с N = 60 и N = 100.

Началната популация е основа за по-нататъшната „еволюция“. Тя се осъществява чрез основните оператори на ГА – селекция, кръстосване и мутация.

Селекцията представлява избор по определен критерий на индивиди от популацията (родители), които ще създадат ново поколение. Критерият, по който ГА може да разбере колко добър е даден индивид (за да бъде избран или не), е фитнес функцията. В нашия случай тя е теоретично определена и представлява общата скорост за извършване на цялото обособено действие: $f = \sum_1^k \omega_i$.

Как се изчислява f? Тъй като за един ключов кадър i, където $i \in [1, k]$, времето за завъртане на всички стави е едно и също, а ъглите са с различни стойности, логично е да се предположи, че времето трябва да е достатъчно за достигането до най-голямата зададена ъглова стойност. Оттук следва, че скоростта ω_i за i–тия ключов кадър е $\omega_i = \frac{\theta_{i_{max}}}{t}$, където $\theta_{i_{max}}$ е най-големият зададен ъгъл за достигане.

Алгоритъмът се стреми да максимизира f, т.е. да повиши скоростта на единичното действие. Това естествено ще доведе до увеличаване на скоростта на ходенето като цяло.

Вероятността p_j един индивид да бъде избран за родител е изчислен на база отношението на приспособимостта му f_j към общата приспособимост $\sum_1^N f_j$: $p_j = \frac{f_j}{\sum_1^N f_j}$, където N е броят на индивидите, участващи в популацията. Това е т.нар. пропорционална стратегия (roulette-selection, Goldberg, 1989), при която колкото по-добър е даден индивид, толкова вероятността да бъде избран е по-голяма.

Размножаването в популацията е извършено стандартно – чрез кръстосване и мутация. За формиране на нови индивиди, двамата родителя са групирани в родителска двойка чрез панмиксия.

Изследвани са три метода на кръстосване – едноточково (one-point crossover), многоточково (uniform crossover) и аритметично (arithmetic crossover) кръстосване. За този домейн оптимално решение се получава при аритметичното кръстосване (таблица 2).

Кръстосване	Min/max поколение	Средна скорост	Max постигната скорост
Едноточково	119/1397	61.0648	80 %
Многоточково	253/18862	60.0861	70 %
Аритметично	299/1059	61.3954	100 %

Таблица 2. Резултати от 10 опита при $f_{max} = 61.3954\ rad/s$, N = 60, P_c = 0.8, P_m = 0.05

Вероятността за кръстосване показва каква част от новото поколение са деца, комбинирали гените на родителите си и каква част са индивиди от предишното поколение. Тя запазва разнообразието в поколенията. Експериментирано е с вероятност за кръстосване е P_c = 0.8 и P_c = 0.9 (таблица 3). Вижда се, че при 80 % вероятност за кръстосване, независимо от големината на популацията, броят поколения е по-голям, но и средната скорост е по-добра.

Популация	Вероятност за кръстосване P_c	Min/max/средно поколения	Средна скорост	Max постигната скорост
N = 60	0.8	299/1059/556	61.3954	100 %
	0.9	173/921/400	60.8964	60 %
N = 100	0.8	250/744/427	61.3954	100 %
	0.9	190/500/274	61.3289	90 %

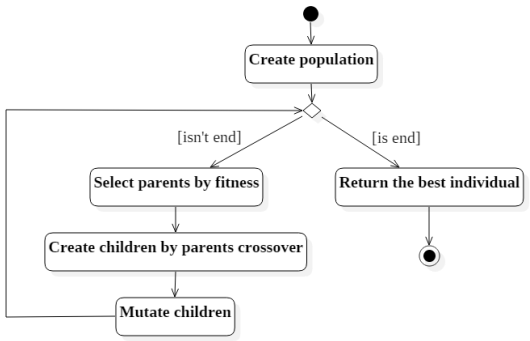
Таблица 3. Резултати от 10 опита с аритметично кръстосване при $f_{max} = 61.3954\ rad/s$, $P_m = 0.05$

За да се осигури запазване на най-добрия индивид от всяко поколение, в допълнение е използван методът на елитарната селекция с размерност 1.

Кръстосването позволява бързо да се намери екстремум. Проблемът е, че този екстремум може да е локален. За да се предпази алгоритъмът от попадане в локален екстремум, е използвана мутация – случайна промяна в един от гените на хромозомата. Тя разширява пространството на търсене, с което вади алгоритъма от локално зацикляне, но вероятността за появяването ѝ е малка (P_m = 0.05), защото иначе търсенето ще се изроди в случайно. За осъществяване на мутацията е използвано добавяне на малко число (± 20) към алелите. Алгоритъмът приключва при достигане на максимална фитнес функция $f_{max} = \sum_1^k \omega_{i_{max}} = \sum_1^k \frac{\theta_{i_{max}}}{t_i}$ или при попадане в „плато“ – достигане на някаква оптимална скорост, която не може да се подобри в рамките на 100 поколения.

На фигура 2 е показана активити диаграмата на реализирания генетичен алгоритъм, който схематично може да се опише по следния начин:

1. Генерира се популация от N-индивида и се изчислява фитнес функцията за всеки един индивид, както и общата фитнес функция за популацията.
2. Селектират се най-добрите индивиди чрез метода на рулетката.
3. Към новото поколение елитарно се прехвърля най-доброто решение.
4. Групираните чрез панмиксия родители аритметично се кръстосват с вероятност P_c.
5. Формираните нови индивиди мутират с вероятност P_m, като променят един от гените си с ± 20.
6. Действия от 2 до 5 се повтарят, докато новата популация изпълни зададеното условие за край – максимум или плато на фитнес функцията.

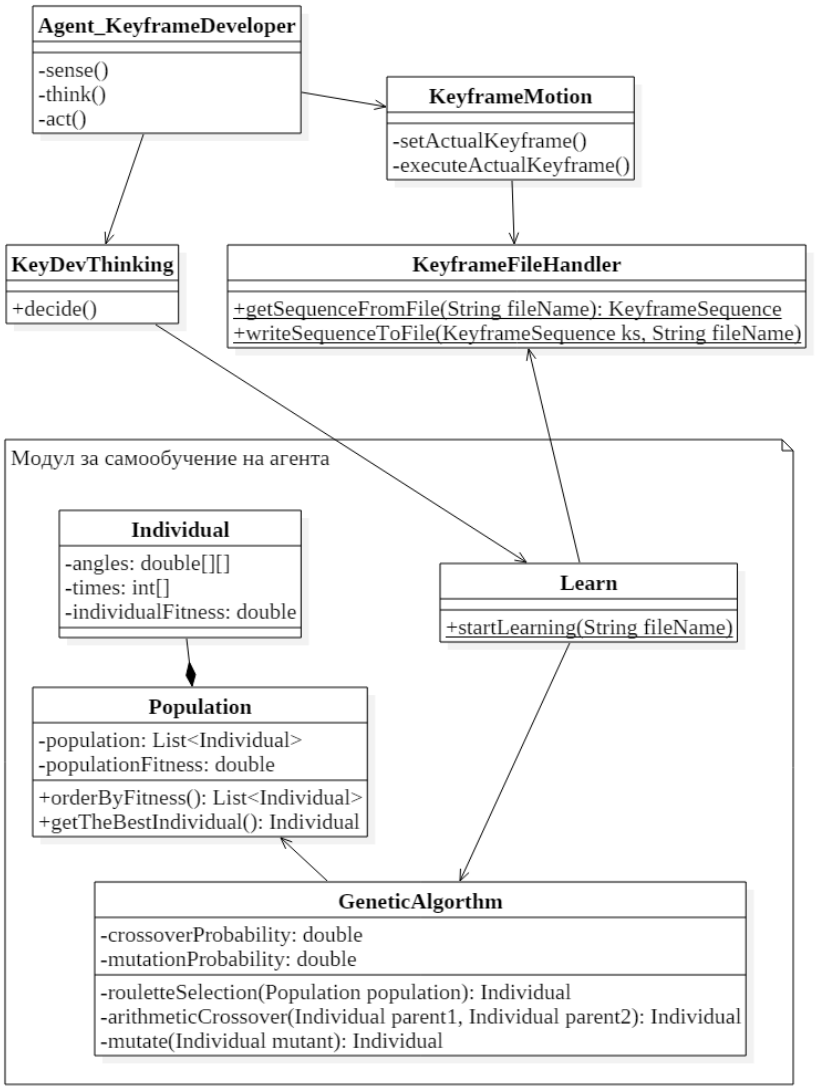


Фигура 2.

Фигура 3 показва основните класове, използвани за реализацията на модула за самообучение на агента на робота.

Агентът (клас Agent_KeyframeDeveloper) изпълнява цикъла „sense-think-act“, като възприема околната среда, обмисля какво и как да направи и действа. Тъй като

роботът е футболист, действията му се свеждат до придвижване по игралното поле, следене на топката и съответно вкарване на гол. Разработеният модул помага на агента да подобри скоростта си на движение.



Фигура 3.

Какво се случва най-общо: в даден момент агентът тръгва да ходи. За целта се прочита текстов файл, съдържащ поредица от необходимите ключови кадри (метод getSequenceFromFile() на класа KeyframeFileHandler). Ако при движението си роботът падне, агентът се обръща към модула за самообучение (метод decide() на класа KeyDevThinking). Методът startLearning() на класа Learn разбира в кое обособено движение е проблема и стартира ГА, за да го подобри. След като ГА намери най-доброто решение (което може), класът Learn го записва в същия текстов файл (метод writeSequenceToFile() на класа KeyframeFileHandler). На следващия цикъл се изпълнява движението с подобрените параметри от новозаписания текстов файл.

Заключение

Реализиран е Java-модул за подобряване скоростта на движение на двукрак робот в симулирана среда. Общото време за развъртане на двигателите на ставите е сведено до оптималното за един цикъл, благодарение на генетичен алгоритъм.

Модулът е имплементиран и тестван с RoboNewbie фреймуърк и SimSpark симулатор и монитор. В процеса на експериментиране са изследвани три метода на кръстосване. Въпреки че разликите не са големи, най-добър резултат се получава при аритметично кръстосване, с големина на популацията N=100, вероятност за кръстосване P_c = 0.8 и вероятност за мутация P_m = 0.05. Приложени са пропорционална и елитарна селекцията на индивидите.

Изследването показа, че ъгловата скорост на ставите може да бъде максимизирана с помощта на този алгоритъм. Това, обаче, което още трябва да се направи, е да се отчете и времето за транслация на кинематичните двойки, тъй като се оказва, че то е по-голямо от времето за развъртане на ставите. Разширяването на алгоритъма в тази посока ще позволи на робота да върви не само бързо, но и стабилно.

Благодарности

Изследването е частично финансирано по проект No. MU17-FMI-001 EXPERT^{L} (Experimental Personal Robot That Learn), 2017-2018.

Литература

Stoyanov, S., A Virtual Space Supporting eLearning, *Proceedings of the Forty Fifth Spring Conference of the Union of Bulgarian Mathematicians Pleven*, April 6–10, 2016, 72–82.

Valkanov, V., S. Stoyanov and V. Valkanova, Virtual Education Space, Journal of Communication and Computer, ISSN:1548-7709 (Print) 1930-1553 (Online), Vol. 13, No. 2, February 2016 (Serial Number 124), 64–76.

Stoynov, S., H. Zedan, E. Doychev, V. Valkanova, A. Stoyanova-Doycheva and V. Valkanov, Context-Aware E-Learning Infrastructure, *The ICT Age* (Eds. A. Ravindran and E. Prakash), Cambridge Scholars Publishing, ISBN (10): 1-4438-8714-5, ISBN (13): 978-1-4438-8714-4, 2016, 221–280.

Doychev, E., A. Stoyanova-Doycheva, S. Stoyanov and V. Ivanova, Agent-Based Support of a Virtual eLearning Space, *Proccedings of 8th International Conference ICCCI 2016* (eds. N. T. Nguyen, Y. Manalopoulos, L. Iliadis, B. Trawinski), Part II, September 28–30 2016, Halkidiki, Greece, Computational Collective Intelligence, Springer Verlag, 35–44.

Stoyanov, S., A. Stoyanova-Doycheva, E. Doychev and K. Gramatova, Virtual Education Space, *The Journal of Applied Science*, Applied Science University, Kongdom of Bahrain, Vol. 1 (1), 2016, 24–40, ISSN 1764-2210.

Стоянов, С., А. Тоскова, Й. Тодоров и Д. Русев, ПРОЕКТ EXPERT^{L}, International Conference "Automatics and Informatics'2017", София, 2017.

Наков, П. и П. Добриков, *Програмиране = ++Алгоритми;*, 2015.

Holland, J., A Mathematical Framework for Studying Learning in Classifier Systems, *Physica D*, 22:307-317, 1986.

Goldberg, D., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

Girardi, N., Ch. Kooijman, A. Wiggers and A. Visser, Automated Optimization of Walking Parameters for the Nao Humanoid Robot, *BNAIC*, Issue 25, 2013, 72–79.

Graf, C. and Th. Röfer, A Closed-loop 3D-LIPM Gait for the RoboCup Standard Platform League Humanoid, *Proceedings of the Fifth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots*, Nashville, TN, USA, 2010.

Zhao, J., L. Qi, J. Yan and Y. Zhu, Dynamic stability gait planning of kid humanoid robot, Guilin, *Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics*, Guilin, China, December 19–23, 2009.

Liu, Ch., D. Wang, E. Goodman and Q. Chen, Adaptive Walking Control of Biped Robots Using Online Trajectory Generation Method Based on Neural Oscillators, *Journal of Bionic Engineering*, Vol. 13, Issue 4, October 2016, 572–584.

Endo, K., F. Yamasaki, T. Maeno and H. Kitan, A Method for Co-Evolving Morphology and Walking Pattern of Biped Humanoid Robot, *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002, USA.

Meriçli, Ç. and M. Veloso, Improving Biped Walk Stability Using Real-Time Corrective Human Feedback, *The 14th annual RoboCup International Symposium*, Singapore, June 25, 2010.

Toskova, A., Evolutionary algorithm in simulated bipedal robot motion, *International Conference „Automatics and Informatics'2017"*, София, 2017.

RoboCup Soccer Server Maintenance Group, SimSpark_RCSS_0.6.7-win32-release3, 2016 (http://simspark.sourceforge.net/wiki/index.php/Main_Page)

Domanska, M., H. Burkhard, AI-Group, Framework RoboNewbie_1.1T, Humboldt University Berlin, Institute of Informatics, 2016 (https://www.naoteamhumboldt.de/en/projects/robonewbie/)

AI-Group, MotionEditor – Adjusted version for use with RoboNewbie, Humboldt-Universität zu Berlin, Department of Computer Science, 2013 (https://www2.informatik.hu-berlin.de/~naoth/RoboNewbie/MotionEditor.pdf)

Burkhard, H., *Lectures in Cognitive Robotics*, Humboldt-Universität zu Berlin, 2016.

Пловдивски университет „П. Хилендарски",
Факултет по математика и информатика
бул. „България" № 236, 4003, Пловдив
имейл: asy@mail.bg

ROBOT WALKING SPEED OPTIMIZATION, IMPLEMENTED BY GENETIC ALGORITHM

Asya Toskova, Stanimir Stoyanov

Summary: *This paper presents an approach to maximize the speed of a humanoid robot walking in a simulated environment by genetic algorithm. The proposed java-algorithm focuses on minimization of joint rotation time.*

Key words: *humanoid bipedal robot Nao, genetic algorithm, fast walking, simulated environment, machine learning, artificial intelligence*