Direct Programming of a Central Pattern Generator for Periodic Motions by Touching

Fabio Dalla Libera [a], Takashi Minato [b], Hiroshi Ishiguro [c,b], Emanuele Menegatti [a]

[a] *Intelligent Autonomous Systems Laboratory, Department of Information Engineering (DEI), Faculty of Engineering, University of Padua, Via Gradenigo 6/a, I-35131 Padua, Italy*

[b] *ERATO, Japan Science and Technology Agency, Osaka University, Suita, Osaka, 565-0871, Japan*

[c] *Department of Adaptive Machine Systems, Osaka University, Suita, Osaka, 565-0871 Japan*

Abstract

Much of the literature shows that Central Pattern Generators (CPGs) are a good approach for generating periodic motions for legged robots. In most of the presented works the numerous CPG parameters are set by automatic techniques like genetic algorithms. This gives the user little control over the resulting motions, since all of the desired features of the motion must be encoded by a fitness/score function. In this paper we present the idea of setting the CPG parameters by interaction with the user, in particular by using tactile interaction. Key elements of the system are a CPG network, a touch protocol and a self-collision prevention system. In this paper we present a practical implementation of each element that confirms the feasibility of the method.

Key words: CPG, human-robot interaction, touch
PACS:

1 Introduction

Many of the rhythmic movements of animals are controlled by central pattern generators (CPGs), i.e. by parts of the nervous system that produce rhythmic activity even in the absence of sensory input. In some cases one of the neurons (called a pace-maker) oscillates on its own, and the other cells oscillate because of the synaptic interactions. In other cases a network of neurons can contain only neurons that do not oscillate on their own but that oscillate when they interact. Living beings and robots have very different physical properties, for instance, while the former are powered by elastic pairs of antagonistic muscles, the latter usually consists of chains of rigid links and joints. However, systems similar to animal CPGs, often realized as weakly coupled oscillators, have been proposed for the control of many kinds of robots, such as hexapods [1], quadrupeds [2,3], bipeds [4–7], snake robots [8], etc. CPGs offer several advantages in terms of simplicity and the ease with which sensory feedback can be introduces [9] as well as in the adaptability and robustness [10] of the system. As reported in [10] the determination of CPG parameters is still difficult because they depend both on the robot and on the environment.

Several methods such as genetic algorithms [11], policy gradient [12] or reinforcement learning [10] have been proposed to automatically set the CPG parameters. Although there are several advantages to these methods, this gives the user little control over the resulting motions, in fact the programmer must express the quality of the motion in terms of a fitness function. For instance, when developing walking motions for humanoid robots the speed can be easily calculated and used for the determination of the quality of the motion, but other characteristics, such as the similarity with human movements of the walking pattern cannot be easily expressed by a mathematical function. In fact, much of the motions for humanoid robots are still developed manually [13,14].

The idea presented in the paper is to combine the use of a CPG with user motion development. Expressly, the motion pattern will be generated by a CPG, whose parameters are set by interaction with the user. Several kinds of human-robot interactions could be used; speech recognition with commands like "move the leg faster" or "raise the knee more" could be employed but this would require much user effort in learning the terms and grammar structures understood by the robot [15]. Learning by watching [16] could also be used, but in this case the complexity of the system would need to be greatly increased. We therefore used tactile interaction, a very powerful method for direct human-robot communication [17].

Using tactile interaction the user is able to directly touch the part whose movement should be modified. We suppose this leads to a very low mental effort. In fact pushing a link in the direction along which its motion should be modified appears to be much more intuitive than considering, for each desired modification, which is the joint along the kinematic chain that is responsible for the movement in that direction. This would be required, for instance, if the CPG parameters were to be set by console or by an interface where the values of the various parameters are changed by sliders. A touch protocol must be defined since with touch the users must be able to express different things, for instance they may need to change the velocity or the range of each joint motion.

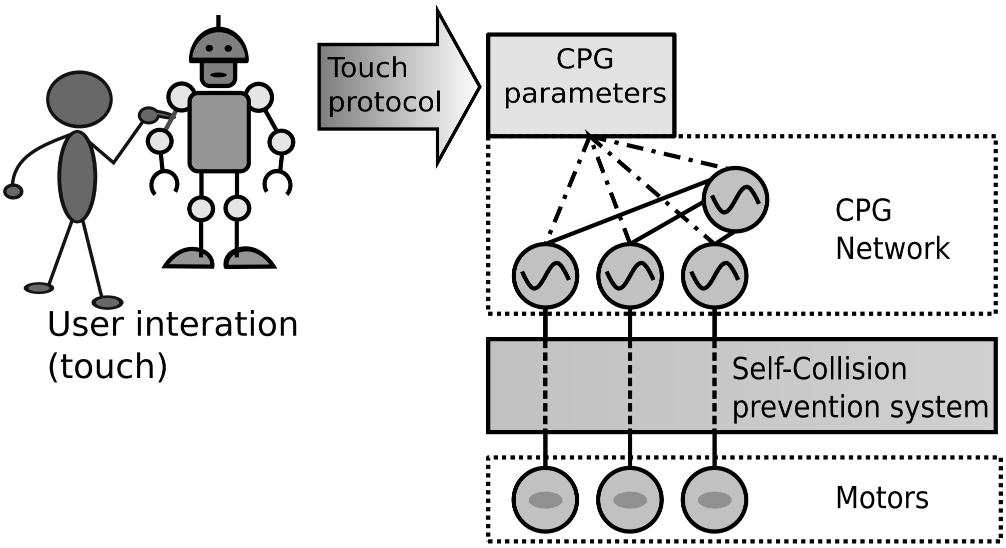


Fig. 1. Basic concept presented in this paper. The motors are controlled by the output of a CPG network, filtered by a system that prevents self collisions of the robot. The user interacts with the robot to modify the CPG parameters and therefore the robot's movement.

Using this touch protocol the user can modify the parameters of the CPG, and therefore the motion, until the movement is considered satisfactory. In many cases the motions obtained during the development process could cause self-collision of the robot. While this could be acceptable in simulation, this should be avoided when using a real robot, especially when this is equipped with servomotors (as for small humanoids robots).

We will present the single elements of the developed system in detail in the following sections (see Figure 1 for a conceptual schema). Section 2 will illustrate the CPG, the touch protocol will be explained in section 3 and the self-collision prevention system will be described in section 4. Section 5 will report an example of the usage of the developed system in the realization of three motions for a humanoid robot. Section 6 will conclude by presenting future works.

2 CPG Network

Many alternatives are available in the choice of the type of the kind of oscillator/neurons employed for the realization of the CPG, for instance sinusoidal [5], Hopf or adaptive frequency Hopf oscillator [6], Rayleigh [7], Van Der Pol [18], FitzHugh-Nagumo [3], Hopfield [19], Hopfield with synaptic depression [20] or Matsuoka's [21].

Among all the possible structures for the connection between the oscillators we can notice that essentially five structures are present in the literature:

(1) chain [22,9,23,24], used mainly for snake robots
(2) star [25,26,5,27,28], that is a “pacemaker”/ “clock” oscillator which provides a synchronizing signal to the other oscillators
(3) tree [29–31], where essentially the oscillators are connected as a tree, from proximal to distal joints
(4) connection between homologous joints [32–40], i.e. joints with a similar function
(5) full connection between the oscillators [41–43]

Our purpose is to let the user modify the behavior of the robot by changing (through touching) the oscillator parameters. We therefore need to have predictable changes in the behavior of the oscillators (i.e. the movement of the robot) consequent to a change in the parameters. For this reason we decided to employ a very easily controllable oscillator network, namely Hopf oscillators connected in a star shape. Expressly each of the n degrees of freedom of the robot is controlled by one oscillator and a further “clock” oscillator provides a reference signals for these oscillators. Let us identify by C_0 the reference oscillator and by C_j, $1 \leq j \leq n$ the oscillators controlling the robot joints,

Using the complex number representation for the Hopf oscillator [44] we have for the j-th oscillator, $0 \leq j \leq n$

$$\begin{aligned} \dot{z_j} &= \gamma \left(\mu_j - |z_j|^2\right) z_j + i\omega_j z_j + F_j(t) \\ m_j &= \Re\left\{z_j\right\} + o_j \end{aligned} \tag{1}$$

where

- $z_j \in \mathbb{C}$ is the state of the oscillator
- $m_j \in \mathbb{R}$ is the control signal for the actuator
- γ is a coefficient for the speed of recovery after perturbation [6]
- $\mu_j \in \mathbb{R}, \mu_j \geq 0$ controls the amplitude of the oscillation
- $\omega_j \in \mathbb{R}, \omega_j \geq 0$ controls the oscillation frequency
- $F_j(t)$ is an external perturbation signal
- o_j is an offset value used to set the position around which the joint oscillates

We decided to restrict the motion set to only include periodic motions, therefore to ensure rational ratios between the frequencies of oscillation of each pair of joints we set

$$\omega_j = p_j \omega_0 \tag{2}$$

$1 \leq j \leq n$, $p_j \in \mathbb{N}$. In the current implementation we do not use any feedback signal, so $F_0(t)$ is zero (the main clock is not influenced by anything, but in further implementation we plan to introduce a signal from the gyroscope) while for $1 \leq j \leq n$

$$F_j(t) = we^{i\phi_j} z_0^{p_j} \tag{3}$$

that is $F_j(t)$ consists essentially in the perturbation from the clock oscillator that permits synchronization of the system. The reference signal z_0 is elevated to the power p_j so the frequencies of the oscillator and of the perturbation are close. Due to the similarity of the frequencies the synchronization is achieved more easily and the phase between the j-th oscillator and the reference one is predictable [45]. The term $e^{i\phi_j}$ allows to alter the phase difference between the oscillators and w is a coupling coefficient between C_0 and the other oscillators. In the current implementation $\mu_0 = 1$,$w = 0.1$ and $\gamma = 1000$.

3 Touch Protocol

As stated in the introduction a protocol mapping touches on the robot sensors to parameter changes must be defined. Suppose for simplicity that each link of the robot is a parallelepiped and that each face constitutes a touch sensor. We need to convert a series of touches on the sensors to CPG parameter changes. Since this work aims exclusively at validating the feasibility of the approach, we decided to simplify the system as much as possible and opted for a static mapping between the user actions and the parameter changes. Analysis of adaptable systems that better fit the user will be conducted in future works. From Equations 1 and 2 it is possible to see that for each joint we can control

- the amplitude of the oscillation of each joint, by μ_j
- the frequency of the movement of each joint, by p_j
- the phase of the movement, with respect to the main oscillator, through the parameter ϕ_j
- the zero position (offset) around which the joint moves, by o_j

and by changing ω_0 we can change the global speed of the movement. We adopted, nearly arbitrarily, the protocol described in the following.

When the user pushes a sensor s as a first step we calculate the most distal joint, along the kinematic chain, that causes a movement of the sensor center in the direction perpendicular to the sensor surface. Intuitively we change the parameters of the oscillator controlling the most distal joint that would move if the joints were not powered and we would push sensor s. More formally suppose the robot's main body (in the case of a humanoid robot, the torso) is fixed in space, and denote by d_j the vector representing the derivative of the position in the space of the center of the pushed sensor when the $j-th$ joint is rotated. Identify by n_s the vector perpendicular to the pushed sensor surface. Denote by $j_1, j_2, ...j_q$ the indices of the q joints that connect the robot's main body to the link where the sensor is located, in order from the most distal to the

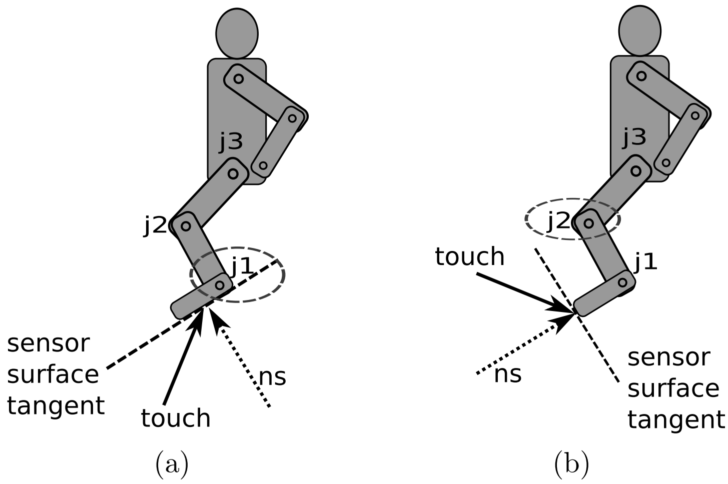


Fig. 2. Example of the determination of the joint whose parameters should be modified when a sensor is pressed. In the first case the parameters of joint j_1 are modified, in the second case, since the $\rho_1 =< n_s, d_{j_1} >$ is zero j_2 is selected.

most proximal (see Fig. 2; we identify the joint j_s such that $\rho_s =< n_s, d_{j_s} > \neq 0$ and $< n_s, d_{j_k} >= 0$ for $s < k \leq q$. If this joint doesn't exist we simply ignore the sensor pressure (unless it is on the main body, as will be specified later).

Once we have determined j_s, i.e. the joint whose parameters should be modified we identify which of the parameters should be modified. For simplicity and robustness of the system we consider the touch sensors as binary and chose the parameter to be modified depending on the pressure pattern over time. In particular we change the offset if the user keeps pushing for a very long time, the amplitude if the pressure lasts a shorter time, the phase if the user operator taps one time and the frequency if two taps are provided (see Fig. 3).

More formally once j_s is determined the phase of joint j_s ($\angle z_{j_s}$) is used as a time reference. In particular the pushing time τ_{j_s} is expressed in terms of difference of phase between the release time and the pushing time, counting for the phase resettings (i.e. the difference is positive, monotonically increasing and can be larger than 2π). We distinguish the following cases:

- If the user pushes for a very long time, $\tau_{j_s} > \Theta_O$ the offset parameter is changed according to the applied force, i.e. $o_{j_s,new} = o_{j_s,old} + sgn(\rho_{j_s})\Delta_O$, where sgn is the sign function.
- If $\Theta_A < \tau_{j_s} \leq \Theta_O$ the amplitude parameter is updated by the value $sgn((\rho_{j_s} * m_{j_s})\Delta_A$ where m_{j_s} is the value of the output at the pushing time.
- If the user pushes for a time $\tau_{j_s} \leq \Theta_A$, releases the sensor and doesn't push it for a time Θ_P then the phase parameter ϕ_{j_s} is updated such that in the following cycles the closest maximum of oscillation occurs at the pushing time, i.e. the quantity $-\angle(m_{j_s} * z_{j_s})$ is added to ϕ_{j_s}, where z_{j_s} and m_{j_s} are considered at the pushing time.

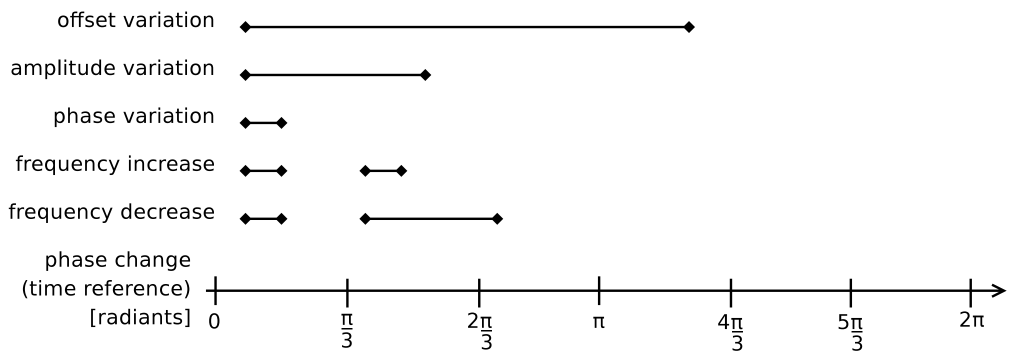


Fig. 3. Schematization of the touch patterns recognized by the system. The presence of a line indicates the pressure of a sensor. The phase change axis (time reference) is also reported.

- If the user pushes for a time $\tau_{j_s} \leq \Theta_A$, releases the sensor and before a phase change of Θ_P pushes the sensor again then p_{j_s} is incremented or decremented respectively if this second pushing time $\tau_{j_s,2}$ is greater or lower than Θ_A.
- Similarly, if the user pushes the robot main body for a time $\Delta\phi_0 \leq \Theta_A$ releases the sensor and before a phase change of Θ_P pushes the sensor again then ω_0 is increased or decreased by the quantity Δ_{ω_0} respectively if the second pushing time is greater or lower than Θ_A.

In our implementation all the Δ and Θ values are constants, expressly $\Theta_O = \pi$, $\Theta_A = \frac{\pi}{6}$, $\Theta_P = \frac{2\pi}{3}$, $\Delta_O = \Delta_A = \frac{\pi}{12}$, $\Delta_{\omega_0} = 1$.

We should note that the user applies forces that interfere with the robot's motion, however this are ignored since we imagine that the user touches the robot while this is moving, waits that the movement stabilizes after the parameter change and then decides the next modification.

4 Self Collision Prevention System

As done in [46] we introduced a system to avoid self collisions. The collision prevention system models each robot link by a set of containing parallelepipeds. A list of couples of parallelepipeds that cannot collide (for instance, two links directly connected by a joint) and for which the computation is skipped is also maintained. For each joint a list of parallelepipeds that are moved (considering the main body fixed) when the joint rotates is also precomputed, so collision detection after the rotation of a single joint can be performed ignoring most of the collision checks. The systems memorizes the current motor positions as an n-dimensional vector $Y_{current}$ and given a vector Y_{target} representing the desired motors position (the outputs o_j of the oscillators) calculates what is the maximum rotation of each joint toward Y_{target} such that there's no collision between the parallelepipeds. More precisely for each joint, in a specified order (the joint ID in the current implementation, see Fig. 4(a)), the system calculates the maximum rotation from the current angle to the target angle, with steps of 3 degrees, that doesn't cause any collision between the links moved by the rotation and the others. The position obtained in this way is sent to the motors and stored in $Y_{current}$. As most of the movements are periodic, collision detection computation is further reduced employing a direct mapped cache: the address is the position of each motor, discretized with granularity of one degree and the content is whether the configuration causes self collisions or no. The mapping function between addresses and cache index is the hash function

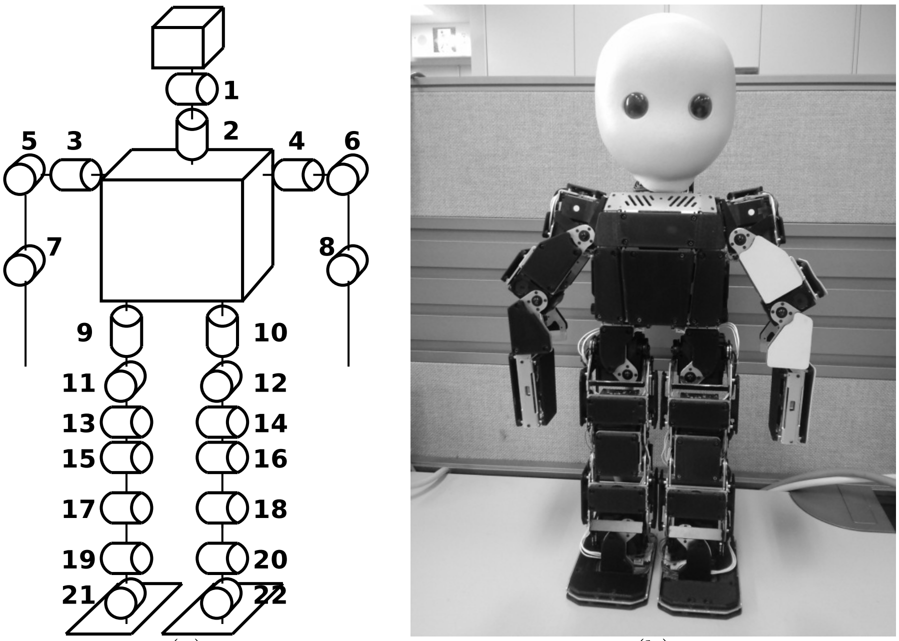
$$f(Y) = mod(\sum_{i=1}^{n} Y_i \varrho_i, S)$$

where Y_i is the position of the i-th joint, ϱ_i is the i-th prime number, S is the cache size and *mod* is the modulo function (in the current implementation $n = 22$ and $S = 27077$).

5 Experiment

As stated in the introduction to validate the feasibility of the approach we developed three motions for a humanoid robot. This was done using a simulator (see [47] for the details) of Vstone's Vision4G. Figure 4(a) reports a schema of its degrees of freedom as well as a photo of the real hardware.

Using a simulator allows us to simplify the discrimination of touches by the user and the pressures due to the gravity, which we need to ignore. In this case mouse clicks were used to simulate pressure applied to the sensors. Starting with $\mu_j = o_j = \phi_j = 0$ and $p_j = 1$, $1 \leq j \leq n$ a crawling motion was developed by a single user in 56 minutes. The user employed 57 amplitude changes, 39 phase changes, 22 offset changes and 2 frequency changes to obtain a satisfactory movement. Figure 9 reports the position sent to three of the joints during the motion development. We can notice the effect of the collision prevention system on joint 7 and the effect of an amplitude parameter change due to user input on joint 5. A side-step movement was then developed in 29 minutes, using 31 amplitude changes, 4 frequency changes, 18 phase changes and 56 offset changes. Finally the CPG parameters were set to make the robot walk in 34 minutes. This required 60 amplitude changes, 15 frequency changes, 28 phase changes and 132 offset changes.



(a) (b)

Fig. 4. Schema of the joints of the robot with their IDs (a) and photo of Vision 4G (b).

Table 1 reports the final values obtained for the CPG parameters for the three motions. The final value for ω_0 is 3.256 for the crawling, 5.256 for the side step and 1.256 for the walking motion. Figures 5, 6 and 7 show screenshots of the three motions. A further tuning, to assure perfect movement symmetry or to maximize the crawling or walking speed could be performed but is beyong the scope of this paper.

For comparison, Fig. 8 reports a crawling movement obtained with a genertic algorithm (population size 20, 60 generations, real value encoding, roulette wheel selection, mutation probability 1). We can see that the algorithm finds a shape for the legs that minimizes the friction with the ground, and uses the head as a support point to proceed forward by large arm movements. While this solution can lead to a good speed, it definitely looks awkward to humans. A difference in the smoothness of the two motions can be deduced quantitatively observing the roll and pitch in the two cases (see Fig 10). The ranges of variation for the roll and the pitch of the robot are 15.4 and 14.5 degrees respectively for the motion obtained by direct interaction with the CPG and 34.7 and 23.9 for the motion optimized by the genetic algorithm.

Table 1
CPG parameter settings obtained for the three motions.

Joint \ Parameters	Crawling				Walking				Side Step			
	μ_j	p_j	ϕ_j	o_j	μ_j	p_j	ϕ_j	o_j	μ_j	p_j	ϕ_j	o_j
1	0	1	0	0	0	1	0	0	0	1	0	0
2	0	1	0	0	0	1	0	0	0	1	0	0
3	0.52	1	0.66	0.78	0.17	1	0.4	-0.7	0.35	1	0	-0.7
4	0.26	1	1.61	-0.52	0	1	0	0.35	0.17	1	1.86	0.17
5	0	1	9.72	0.52	0	1	0	1.22	0	1	0	1.05
6	0.52	1	2.31	0	0	1	0	-1.05	0.35	1	4.4	-1.4
7	0.26	1	2.20	0.78	0.17	1	0.49	-0.17	0	1	0	0
8	0.78	1	5.69	-0.52	0.17	1	0	-0.35	0	1	5.5	0
9	0	1	0	0	0	1	0	0	0	1	0	0
10	0	1	0	0	0	1	0	0	0	1	0	0
11	0	1	5.68	0	0	1	1.64	0	0	1	0	0
12	0	1	0	0	0	1	0	0	0.17	1	0	0
13	0	1	0	0	0	1	0	0	0	1	0	0
14	0	1	0	0	0	1	0	0	0	1	0	0
15	0.26	1	1.28	-1.05	0	1	0	-0.52	0	1	0	0
16	0.26	1	0	1.05	0.17	1	0.31	0.35	0	1	0	0
17	0.26	1	0	0	0	1	0	0.52	0	1	0	0
18	0.26	1	0	0	0	1	0	-0.52	0	1	0	0
19	0	1	0	0	0	1	0	0	0	1	0	0
20	0	1	0	0	0	1	0	0	0	1	0	0
21	0	1	0	0	0	1	0	0	0	1	0	0
22	0	1	0	0	0	1	0	0	0	1	0	0

6 Conclusions and Future works

In this paper tactile interaction was proposed as a way to set the numerous parameters of a CPG network. The main components of the practical implementation were described and as a validation a crawling motion, a side stepping motion and a walking gait were developed from scratch using the proposed approach. We stress that purpose of the experiment is not to achieve the fastest locomotion speed which is possible but to achieve a motion satisfactory for the user (in terms, for instance, of similarity to human movements). Obviously the motions obtained with this approach could be used as initial solutions to be optimized with classical automatic methods. Future work will involve the discrimination between user touches, self touches and pressures due the environment, a problem here avoided employing a simulator but that must be tackled to employ the proposed approach directly with a real robot. The actual implementation does not include feedback from the sensors or the gyroscope. Since we can expect entrainment with the environment and therefore a better stability and motion variation, future research will aim at including feedback without reducing the approach generality. Furthermore since the trajectories generated by Hopf oscillators are essentially sinusoidal, to broaden the possible movement repertories (for example, keeping the stance leg still while walking) other types of easily predictable oscillators will be considered.

7 Acknowledgements

The authors would like to thank RC Keely for the comments on the first draft of the paper.

References

[1] M. A. Lewis, A. H. Fagg, G. Bekey, Genetic algorithms for gait synthesis in a hexapod robot, Zheng, ed. Recent Trends in Mobile Robots (1994) 317–331.

[2] T. Ishii, S. Masakado, K. Ishii, Locomotion of a quadruped robot using cpg, in: 2004 IEEE Int. Joint Conf. on Neural Networks, Budapest, Hungary, Vol. 4, 2004, pp. 3179–3184.

[3] J. J. Collins, S. A. Richmon, Hard-wired central pattern generators for quadrupedal locomotion, Biological Cybernetics 71 (5).

[4] G. Taga, H. Yamaguchi, Y. Shimizu, Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment, Biological Cybernetics 65 (1991) 147–159.

[5] J. Morimoto, G. Endo, J. Nakanishi, S.-H. Hyon, G. Cheng, D. C. Bentivegna, C. G. Atkeson, Modulation of simple sinusoidal patterns by a coupled oscillator model for biped walking, in: 2006 IEEE International Conference on Robotics and Automation (ICRA 2006), Orlando, USA, 2006, pp. 1579–1584.

[6] L. Righetti, A. Ijspeert, Programmable central pattern generators: an application to biped locomotion control, in: 2006 IEEE International Conference on Robotics and Automation (ICRA 2006), Orlando, USA, 2006, pp. 1585–1590.

[7] A. C. de Pina Filho, M. S. Dutra, L. S. C. Raptopoulos, Modeling of a bipedal robot using mutually coupled rayleigh oscillators, Biological Cybernetics 92 (1) (2005) 1–7.

[8] A. Crespi, A. Badertscher, A. Guignard, A. Ijspeert, Amphibot i: An amphibious snake-like robot, Robotics and Autonomous Systems 50 (4) (2005) 163–175.

[9] A. J. Ijspeert, A. Crespi, Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model, in: 2007 IEEE

International Conference on Robotics and Automation (ICRA 2007), Roma, Italy, 2007, pp. 262–268.

[10] Y. Nakamura, T. Mori, M. aki Sato, S. Ishii, Reinforcement learning for a biped robot based on a cpg-actor-critic method, Neural Networks 20 (6).

[11] H. Inada, K. Ishii, Behavior generation of bipedal robot using central pattern generator(cpg) (1st report: Cpg parameters searching method by genetic algorithm), in: 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS 2003), Vol. 3, Las Vegas, USA, 2003, pp. 2179–2184.

[12] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, G. Cheng, Learning cpg-based biped locomotion with a policy gradient method: Application to a humanoid robot, International Journal of Robotics Research 27 (2) (2008) 213–228.

[13] T. Wama, M. Higuchi, H. Sakamoto, R. Nakatsu, Realization of tai-chi motion using a humanoid robot, in: R. Jacquart (Ed.), IFIP Congress Topical Sessions, Kluwer, 2004, pp. 59–64.

[14] J. Baltes, S. McCann, J. Anderson, Humanoid robots: Abarenbou and daodan, in: RoboCup 2006 - Humanoid League Team Description, Bremen, Germany, 2006.

[15] C. Fugen, H. Holzapfel, A. Waibel, Tight coupling of speech recognition and dialog management dialog-context dependent grammar weighting for speech recognition, in: Proceedings of the ICSLP, Jeju-Islands, Korea, 2004.

[16] Y. Kuniyoshi, M. Inaba, H. Inoue, Learning by watching: extracting reusable task knowledge from visual observation of human performance, IEEE Transactions on Robotics and Automation 10 (6) (1994) 799–822.

[17] R. Voyles, P. Khosla, Tactile gestures for human/robot interaction, in: 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS 1995), Pittsburg, USA, 1995, pp. 7–13.

[18] P. Veskos, Y. Demiris, Developmental acquisition of entrainment skills in robot swinging using van der pol oscillators, Proc. of the Epigenetic Robotics 2005, Japan (2005) 87–93.

[19] B. Mathayomchan, R. D. Beer, Center-crossing recurrent neural networks for the evolution of rhythmic behavior, Neural Computation 14 (2002) 2043–2051.

[20] A. L. Taylor, G. W. Cottrell, W. B. Kristan, Analysis of oscillations in a reciprocally inhibitory network with synaptic depression, Neural Computation 14 (2002) 561–581.

[21] K. Matsuoka, Sustained oscillations generated by mutually inhibiting neurons with adaptation, Biological Cybernetics 52 (1985) 367–376.

[22] A. H. Cohen, Control principles for locomotion - looking toward biology, in: Proc. 2nd International Symposium on Adaptive Motion in Animals and Machines (AMAM 2003), Kyoto, Japan, 2003, pp. 41–51.

[23] K. Inoue, S. Ma, C. Jin, Neural oscillator network-based controller for meandering locomotion of snake-like robots, in: 2004 IEEE International Conference on Robotics and Automation (ICRA 2004), New Orleans, USA, 2004, pp. 5064–5069.

[24] T. Williams, Phase coupling by synaptic spread in chains of coupled neuronal oscillators, Science 258 (5082) (1992) 662–665.

[25] S. Schaal, S. Kotosaka, D. Sternad, Nonlinear dynamical systems as movement primitives, in: IEEE-RAS 1st International Conference on Humanoid Robots (Humanoids 2000), Cambridge, USA, 2000, pp. 117–124.

[26] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, M. Kawato, A framework for learning biped locomotion with dynamical movement primitives, in: IEEE-RAS 4th International Conference on Humanoid Robots (Humanoids 2004), Santa Monica, Los Angeles, USA, 2004, pp. 925–940.

[27] S. Aoi, K. Tsuchiya, K. Tsujita, Turning control of a biped locomotion robot using nonlinear oscillators, in: 2004 IEEE International Conference on Robotics and Automation (ICRA 2004), New Orleans, USA, 2004, pp. 3043–3048.

[28] M. A. Lewis, F. Tenore, R. Etienne-Cummings, Cpg design using inhibitory networks, in: 2005 IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain, 2005, pp. 3682–3687.

[29] K. Endo, F. Yamasaki, T. Maeno, H. Kitano, A method for co-evolving morphology and walking pattern of biped humanoid robot, in: 2002 IEEE International Conference on Robotics and Automation (ICRA 2002), Washington, USA, 2002, pp. 2775–2780.

[30] S. Degallier, C. P. Santos, L. Righetti, A. Ijspeert, Movement generation using dynamical systems: a humanoid robot performing a drumming task, in: IEEE-RAS 6th International Conference on Humanoid Robots (Humanoids 2006), Genoa, Italy, 2006.

[31] L. Jalics, H. Hemami, Y. F. Zheng, Pattern generation using coupled oscillators for robotic and biorobotic adaptive periodic movement, in: 1997 IEEE International Conference on Robotics and Automation (ICRA 1997), Vol. 1, Albuquerque, USA, 1997, pp. 179–184.

[32] P. Roy, Y. Demiris, Analysis of biped gait patterns generated by van der pol and rayleigh oscillators under feedback, in: Proc. 3rd International Symposium on Adaptive Motion in Animals and Machines (AMAM 2005), Ilmenau, Germany, 2005.

[33] M. Yano, Purposive locomotion of insect in indefinite environment, in: Proc. 2nd International Symposium on Adaptive Motion in Animals and Machines (AMAM 2003), Kyoto, Japan, 2003, pp. 31–40.

[34] H. Kimura, Y. Fukuoka, Biologically inspired dynamic walking of a quadruped robot on irregular terrain - adaptation at spinal cord and brain stem -, in: International Symposium on Adaptive Motion in Animals and Machines (AMAM 2000), Montreal,Canada, 2000.

[35] H. Kimura, Y. Fukuoka, Y. Hada, K. Takase, Three-dimensional adaptive dynamic walking of a quadruped - rolling motion feedback to cpgs controlling pitching motion, in: 2002 IEEE International Conference on Robotics and Automation (ICRA 2002), Washington, USA, 2002, pp. 2228–2233.

[36] K. Nakada, T. Asai, Y. Amemiya, An analog cmos circuit implementing cpg controller for quadruped walking robot, in: Proc. 2nd International Symposium on Adaptive Motion in Animals and Machines (AMAM 2003), Kyoto, Japan, 2003.

[37] H. Takemura, J. Ueda, Y. Matsumoto, T. Ogasawara, The influence of friction on gait and energy efficiency of the walking robot based on rhythmic control, in: Proc. 2nd International Symposium on Adaptive Motion in Animals and Machines (AMAM 2003), Kyoto, Japan, 2003.

[38] G. Endo, J. Morimoto, J. Nakanishi, G. Cheng, An empirical exploration of a neural oscillator for biped locomotion control, in: 2004 IEEE International Conference on Robotics and Automation (ICRA 2004), New Orleans, USA, 2004, pp. 3036–3042.

[39] J. S. Bay, H. Hemami, Modeling of a neural pattern generator with coupled nonlinear oscillators, IEEE Trans. on Biomedical Engineering 34 (4) (1987) 297–306.

[40] M. Cao, A. Kawamura, A design method of neural oscillatory networks for generation of humanoid biped walking patterns, in: 1998 IEEE International Conference on Robotics and Automation (ICRA 1998), Leuven, Belgium, 1998, pp. 2357–2362.

[41] K.Tsujita, K.Tsuchiya, A.Onat, Decentralized autonomous control of a quadruped locomotion robot, Artificial life and robotics 5 (2003) 152–158.

[42] S. Miyakoshi, G. Taga, Y. Kuniyoshi, A. Nagakubo, Three dimensional bipedal stepping motion using neural oscillators–towards humanoid motion in the real world., in: 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS 1998), Victoria, Canada, 1998, pp. 84–89.

[43] W. Ilg, J. C. Albiez, H. Jedele, K. Berns, R. Dillmann, Adaptive periodic movement control for the four legged walking machine bisam, in: 1999 IEEE International Conference on Robotics and Automation (ICRA 1999), Detroit, USA, 1999, pp. 2354–2359.

[44] A. Kern, R. Stoop, Nonlinear dynamics of cochlear information processing, Proc. Int. Workshop of Nonlinear Dynamics of Electronic Systems (NDES-04), Evora, Portugal (2004) 190–193.

[45] K. J. Pikovsky A., Rosenblum M., Synchronization: A Universal Concept in Nonlinear Science, Cambridge University Press, Cambridge, 2001.

[46] S. Degallier, L. Righetti, L. Natale, F. Nori, G. Metta, A. Ijspeert, A modular bio-inspired architecture for movement generation for the infant-like robot iCub, in: 2nd IEEE RAS / EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob), 2008.

[47] F. Dalla Libera, T. Minato, H. Ishiguro, E. Pagello, E. Menegatti, Developing robot motions by simulated touch sensors, SIMPAR 2008, LNAI 5325 (2008) 242–253.

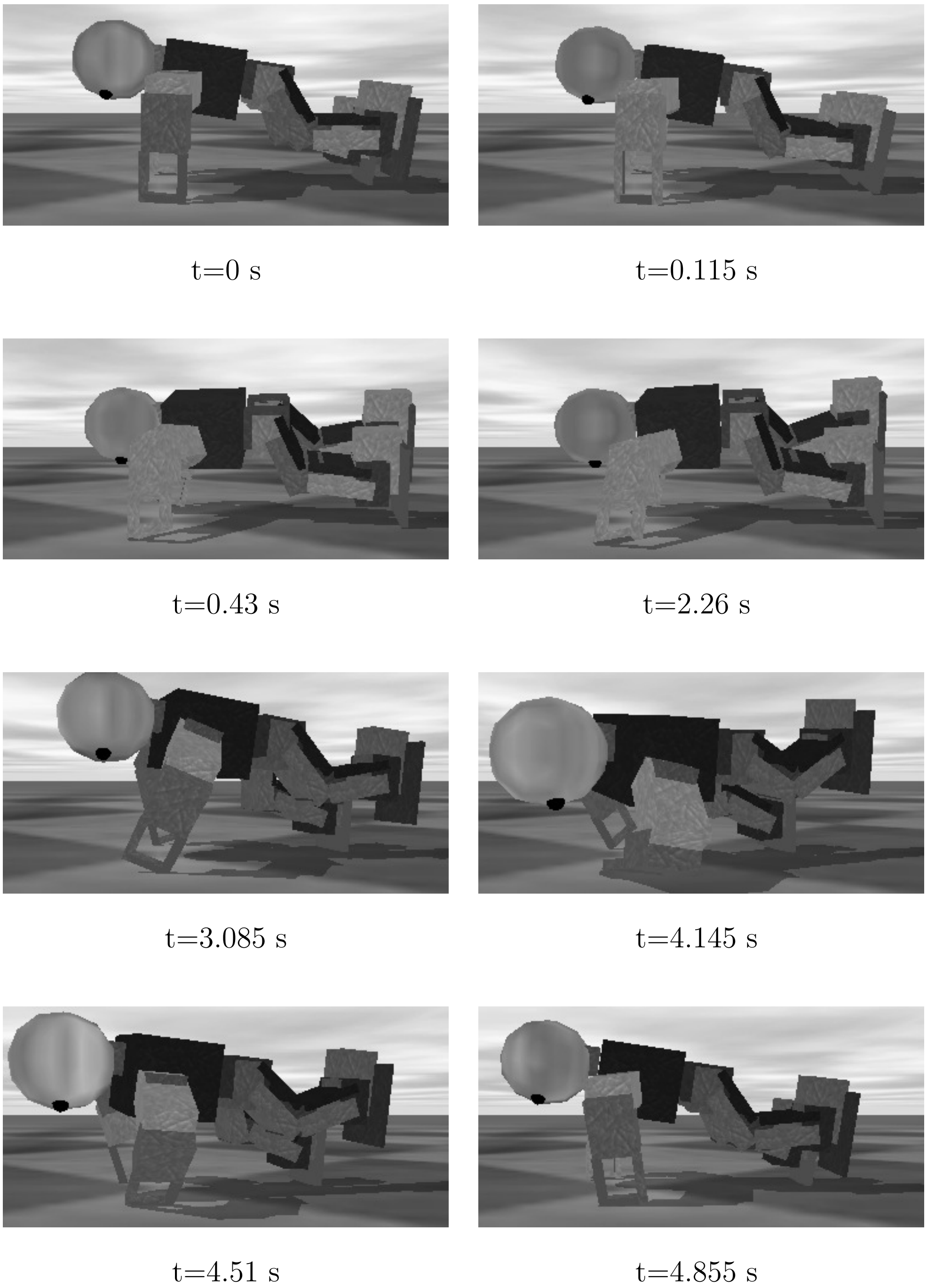


Fig. 5. Execution of the crawling movement.

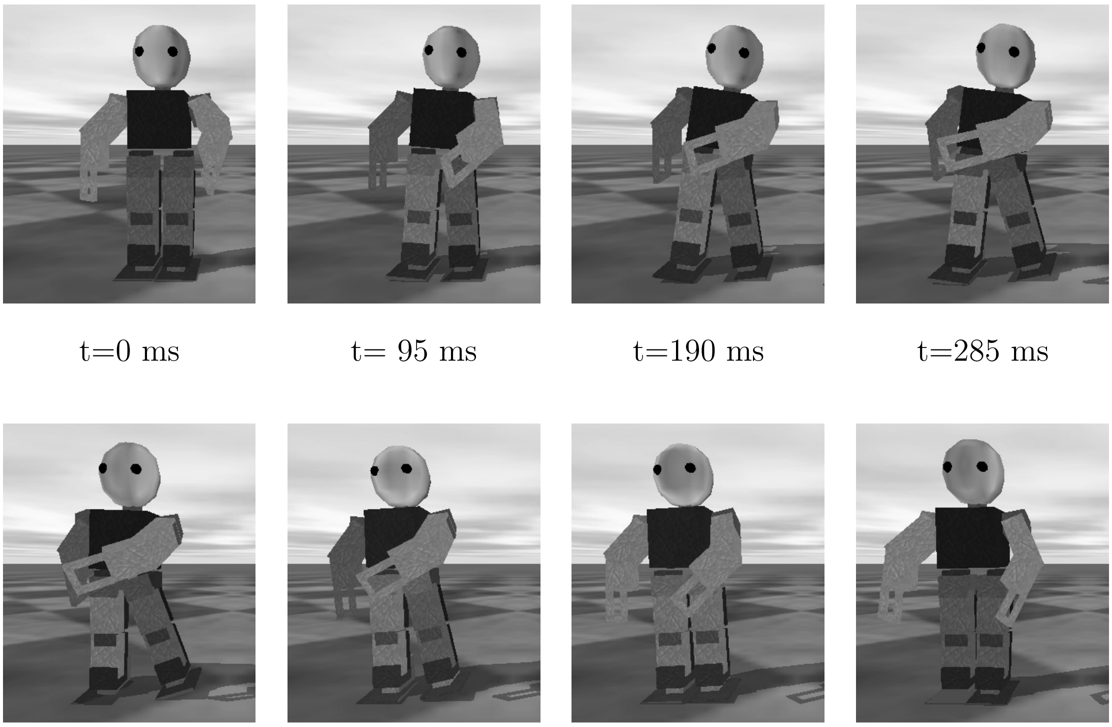
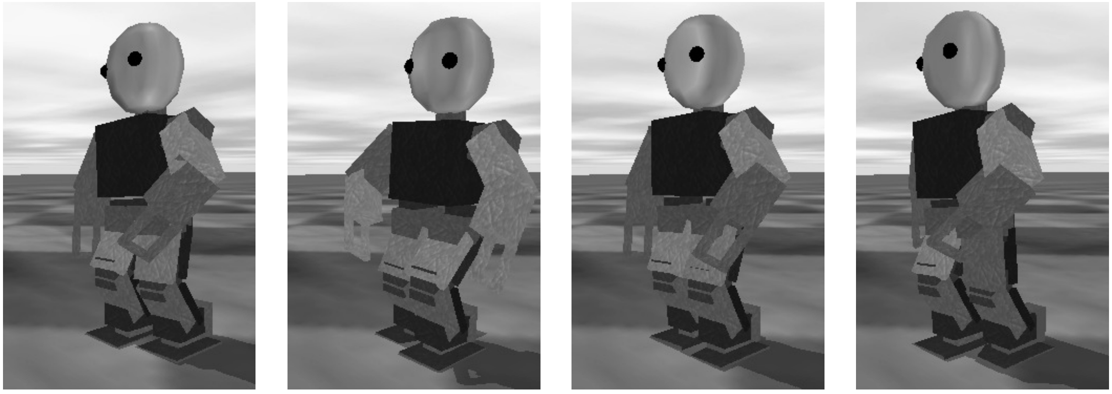
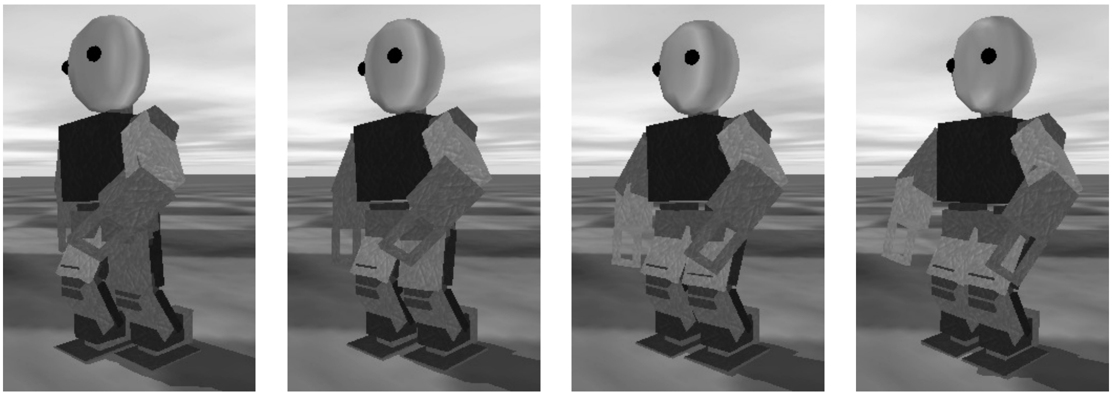


Fig. 6. Execution of the sidestepping movement.



t=0 s t=1.695 s t=2.7 s t=3.488 s



t=4.135 s t=4.945 s t=5.305 s t=5.605 s

Fig. 7. Execution of the walking movement.

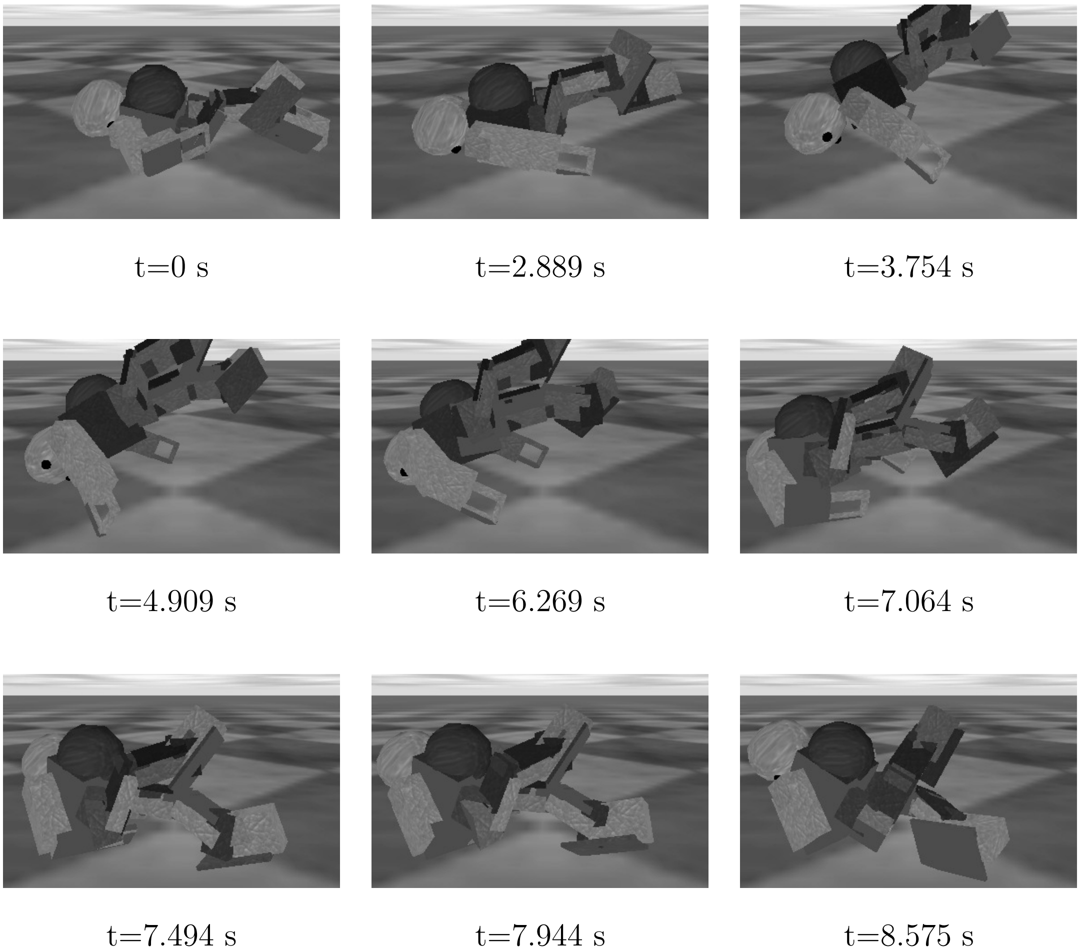


Fig. 8. Execution of a crawling movement obtained with a genetic algorithm.

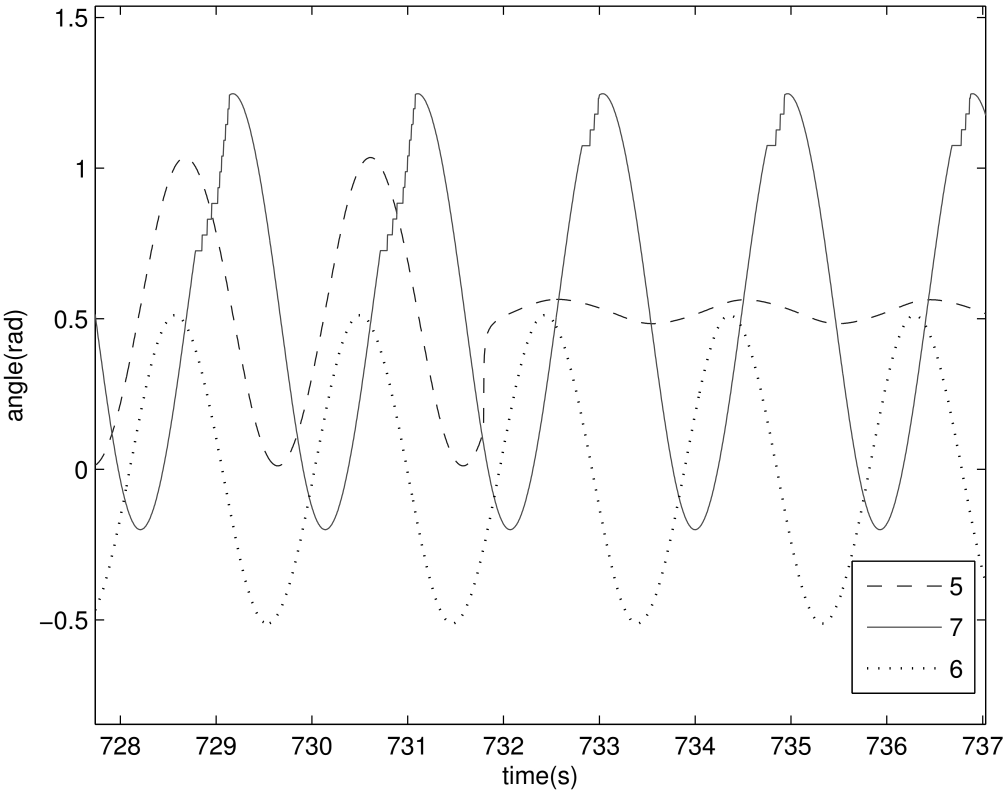


Fig. 9. Evolution of the command signal sent to three of the joints during the motion development. The effect of the self-collision prevention system can be seen for joint 7, while a smooth modification of the oscillation due to a parameter change is easily observable for joint 5.

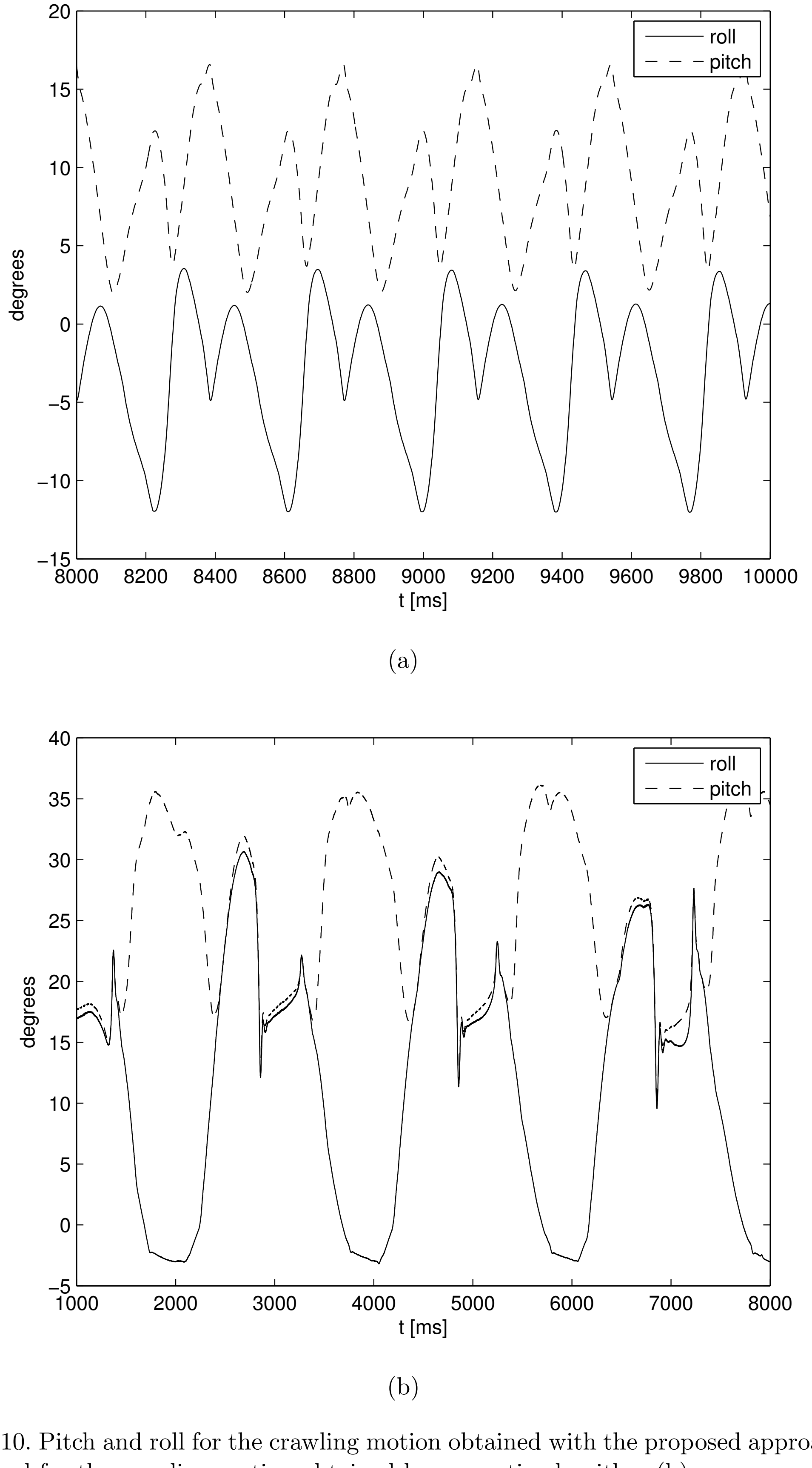
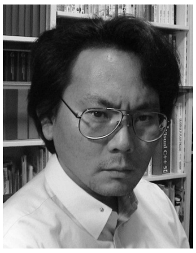


Fig. 10. Pitch and roll for the crawling motion obtained with the proposed approach (a) and for the crawling motion obtained by a genetic algorithm (b).

8 Bibiographies

Fabio Dalla Libera received his Master degree in Computer Science from Padua University, Padua, Italy, in 2007. He is now a Ph.D. student at Padua University and he is studying touch as a means to communicate with humanoid robots. Until now, he has visited Professor Ishiguro's Intelligent Robotics Laboratory at Osaka University four times.

Takashi Minato obtained his Ph.D. degree in engineering from Osaka University in 2004. He was a researcher of CREST, JST since December 2001, and he was an assistant professor of the department of Adaptive Machine Systems, Osaka University since September 2002. He has been a researcher of ERATO, JST since June 2006.



Hiroshi Ishiguro (Member, IEEE) received the D.Eng. degree from Osaka University, Osaka, Japan, in 1991. In 1991, he started working as a Research Assistant of the Department of Electrical Engineering and Computer Science, Yamanashi University, Yamanashi, Japan. Then he moved to the Department of Systems Engineering, Osaka University, as a Research Assistant in 1992. In 1994, he was an Associate Professor, Department of Information Science, Kyoto University, Kyoto, Japan, and started research on distributed vision using omnidirectional cameras. From 1998 to 1999, he worked in the Department of Electrical and Computer Engineering, University of California, San Diego, as a Visiting Scholar. In 2000, he moved to the Department of Computer and Communication Sciences, University, Wakayama, Japan, as an Associate Professor and became a Professor in 2001. He is now a Professor of the Department of Adaptive Machine Systems, Osaka University, and a Group Leader at ATR Intelligent Robotics and Communication Laboratories, Kyoto. Since 1999, he has also been a Visiting Researcher at ATR Media Information Science Laboratories, Kyoto.

Emanuele Menegatti graduated in Physics in 1998 at the University of Padua, Italy. From June 2002 to September 2002, he was Visiting Researcher at the Intelligent Robotics Laboratory of Prof. Hiroshi Ishiguro in the University of Wakayama. He received the Ph.D. in Informatics and Industrial Electronics in 2003 at the Dept. of Information Engineering of the University of Padua. From May 2003 to Dec.2004, he had a Post-Doc Position at the Dept. of Information Engineering of the University of Padua. In January 2004, he was a visiting researcher at the BORG Lab (Georgia Institute of Technology , Atlanta U.S.A.) working with prof. Frank Dellaert. In February-March 2004, he was visiting researcher at the Intelligent Robotics Laboratory of Prof. Hiroshi Ishiguro at Osaka University. In 2006-2007 he became adjunct Professor of Robotics at Padua University.