Interactive Design of Expressive Locomotion Controllers for Humanoid Robots

Sébastien Dalibard, Daniel Thalmann, Nadia Magnenat-Thalmann

2012 IEEE RO-MAN: The 21st IEEE International Symposium on
Robot and Human Interactive Communication.
September 9-13, 2012. Paris, France.

Interactive Design of Expressive Locomotion Controllers for Humanoid Robots

Sébastien Dalibard Daniel Thalmann Nadia Magnenat-Thalmann

Abstract— **This paper presents an interactive dynamic controller used to generate locomotion patterns for humanoid robots. The purpose of this work is to provide animators and artists easy and intuitive tools to design expressive motions for humanoid robots. A review of similar work in the computer animation community has guided our choices regarding the implementation and level of interaction between the user and an inverse dynamics solver. We have used our controller on a model of the Aldebaran humanoid robot Nao, and have generated a few expressive locomotion patterns that are presented in the experimental section of this paper.**

I. INTRODUCTION

Generating believable and expressive motions for social robots remains an open problem. Robots that interact with humans have to convey their affective state, emotions and moods, through different modalities, including speech, touch, facial expressions, body postures and gestures [1], [2]. When controlling social humanoid robots, or any legged robot, the need for expressive motion often interferes with complex stability and balance constraints. For this reason, humanoid motion generation is traditionally taken care of by specialists of robot control through specific interfaces that require a lot of know-how.

On the other hand, in the computer animation entertainment industry, generation of expressive and believable motion for anthropomorphic characters is usually carried out by animators, who do not necessarily have the engineering skills to design motion controllers. In that respect, providing intuitive, effective and flexible software and interfaces to motion artists is one of the key challenges in computer animation research. Our goal is to design similar interfaces that could be used for humanoid robot whole-body motion design. Fig. 1 shows some examples of expressive walking styles applied to a small humanoid robot.

Recently, some joint-space solutions have been proposed to design humanoid robots animations [3]. The definition and edition of motion is done by inputing key-poses in the robot configuration space. This is also the traditional method used by 3D animators. However, when dealing with complex kinematic chains, the process of defining every joint angle for every key-pose can get very tedious. Modifying existing motions, or adapting them to new characters or situations is difficult and time-consuming. Another possibility to generate believable whole-body anthropomorphic motions is to use motion capture systems and retarget captured data onto the animated character or humanoid robot [4], [5]. Because of the differences between human and humanoid robot kinematics and dynamics, the retargeted motion can look very different from the initial captured motion. Moreover, animators do not necessarily have access to a motion capture system. Techniques were proposed in computer animation to generate new motion from existing data [6], [7], by interpolation or extrapolation, but they do not seem well suited for humanoid robotics.

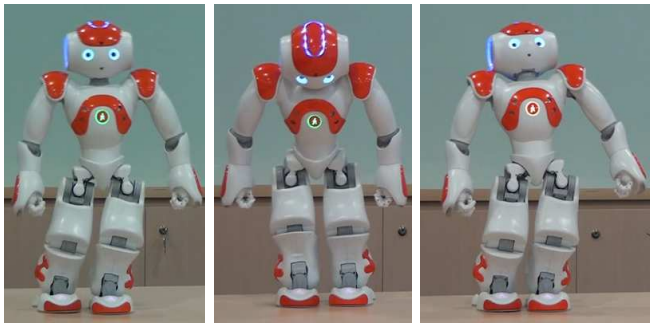

Fig. 1. A humanoid robot walking with three different styles. From left to right: relaxed, sad and proud.

One solution to overcome the limitations of joint-space approaches is to define control in characters' operational space [8]–[10]. This is well studied in both robotics and computer animation literature, for a recent survey, one can refer to [11]. In robotics, operational space control approaches are typically used to solve manipulation task, as well as balance and stability constraints. This paper deals with the design and use of operational-space controllers to design expressive locomotion patterns for humanoid robots. The features that we propose to control are high level properties of the robot dynamic state, such as Center of Mass (CoM) position or trajectory, angular-momentum, end-effector positions, or joint torque minimization. Our choice of features come from similar work [12] from computer animation literature, and recent user studies about the parameters that influence how people perceive virtual characters' affective state during locomotion animations [13]. A small number of physically relevant and intuitive features have been shown to provide enough flexibility to the animator, in order to define different types of stylistic, expressive locomotion patterns.

Next section reviews the related literature and states the contribution of our work. Section III describes the architecture of our locomotion controller, Section IV lists the features that the user can control to define new locomotion patterns

S. Dalibard and D. Thalmann are with Nanyang Technological University, Singapore sdalibard@ntu.edu.sg

N. Magnenat-Thalmann is with Nanyang Technological University, Singapore and MIRALab, University of Geneva, Switzerland

and presents the corresponding user interface, Section V details some implementation choices, Section VI shows experimental results obtained on a model of the Nao humanoid robot and finally Section VII concludes and discusses the limits and potential improvements of our method.

II. RELATED WORK AND CONTRIBUTION

This paper presents a study on the use of computer animation tools for humanoid robot locomotion design. We will thus review the relevant literature from both computer animation and humanoid robotics communities.

A. Computer Animation

Many existing locomotion controllers proposed in the computer animation literature use joint-space representations of motions. They often use per-joint proportional-derivative servo, coordinated by higher level state machines [14]–[16]. Another trend to define stylistic motions is data-driven methods, which interpolate or extrapolate captured data to generate new animations, see for example [6], [7]. These methods produce highly realistic animations, but they are not well suited to generate motions far from the recorded trajectories, or adapted to very different kinematic trees, like a humanoid robot.

The need to interactively adapt motion to unexpected events or to different character models has driven computer animation research towards physically based locomotion controllers. For example, Simbicon [17] uses a joint space representation of locomotion patterns, that adapts to changes in character or environment by physical simulation and control. More recently, in [12], de Lasa et al present dynamic feature-based locomotion controllers, that represent locomotion patterns only in terms of high-level dynamic features. Our approach is quite similar to this last one. Our long-term goal is to make humanoid robots as believable in their motion as state-of-the-art characters in computer animation.

B. Humanoid Robotics

Using operational-space based, dynamic controllers for humanoid robots is a fruitful ongoing research trend [10], [18], [19]. Recent contributions focus on how to handle generic contacts, balance or task prioritization. Our purpose is not to present a new algorithm for dynamic humanoid control, but instead, to use existing methods for expressive anthropomorphic motion design. To our knowledge, this has not been investigated as such in the robotics community. Fast dynamic controllers have recently been used to imitate captured human motion [20]. In our work, we have focused on the use of controllable dynamic features to define new locomotion patterns, and thus do not rely on captured data.

C. Contribution

This paper presents a first try at easing the design of expressive motions for humanoid robots. It is based on ideas from computer animation, that have not been used in social robotics yet. By using dynamic control, we generate physically feasible motions that can be directly applied to real robots. Our main goal is to provide animators – who are not necessarily robotics engineers – with flexible and powerful motion design software, so that robot motions can get as believable and expressive as state-of-the-art computer animated characters.

III. LOCOMOTION CONTROLLER DESCRIPTION

Our controller is used to generate physically possible motion, while satisfying user controlled tasks that define stylistic locomotion patterns. At time t, the robot is in a configuration $q(t)$ and in a state $(q(t), \dot{q}(t))$. In the following, we will ommit the dependance in t. At each instant of simulation and control, the controller determines the following vector of unknowns:

$$u = (\ddot{q}, \tau, \phi)$$

where $\ddot{q}$ is the second derivative of the robot configuration, τ is the vector of actuated joint torques and ϕ is the vector of contact forces applied by the environment on the robot.

Each step of control consist in solving a Quadratic Program (QP). It optimizes motion objectives, which include user controlled tasks, by minimizing a quadratic function of u:

$$\frac{1}{2}u^T Q u + c^T u, \tag{1}$$

while enforcing physical constraints of the form:

$$\begin{aligned} Au &\leq b \\ Eu &= d \end{aligned} \tag{2}$$

The rest of the section details the definitions and computations of Q, c, A, b, E and d. We start by presenting the physics-based constraints, then motion objectives.

A. Physics-based Constraints

1) Dynamic Equation of the system: The dynamic equation describing the motion of an articulated tree of rigid bodies in contact with its environment is:

$$H(q)\ddot{q} + p(q, \dot{q}) = S^T \tau + J_C^T \phi \tag{3}$$

where H is the whole-body inertia matrix, p is the vector of nonlinear effects, including centrifugal, Coriolis and gravity forces, S is a matrix selecting the actuated joints, and J_C is the jacobian that maps joint velocities to world space velocities at the contact points between the robot and the environment. For an extensive description of how to compute these different quantities, the reader can refer to [21]. Eq. (3) is of the form ($Eu = d$) in eq. (2) and is bound to be respected at each step of control.

2) Contacts: At each contact point, the contact force ϕ is expressed in the basis of the linearised friction cone [22], and the force along the normal of the contact surface is positive or null. We found that enforcing static contact points could lead to infeasible QP problems. Instead, we only use non-penetration constraints. Noting $\mathbf{n}$ the contact surface normal at contact C, we enforce the inequality:

$$\mathbf{n}.\left(J_c\ddot{q} + \dot{J}_c\dot{q}\right) \geq 0 \tag{4}$$

Eq. (4) is of the form ($Au \leq b$) in eq. (2).

3) Joint Limits: Joint-limit constraints are handled by the solver by correctly bounding joint accelerations. Noting the duration between two control steps ΔT, for a joint value $q^{(i)}$ bounded between $\underline{q}^{(i)}$ and $\overline{q}^{(i)}$, we enforce:

$$\underline{q}^{(i)} \leq q^{(i)} + \Delta T \dot{q}^{(i)} + \frac{\Delta T^2}{2} \ddot{q}^{(i)} \leq \overline{q}^{(i)} \tag{5}$$

4) Torque Limits: Torque limit constraints are enforced so that generated motions respect robots' physical capabilities. They are of the form:

$$\underline{\tau}^{(i)} \leq \tau^{(i)} \leq \overline{\tau}^{(i)} \tag{6}$$

B. Motion Objectives

We have presented the constraints enforced by the QP solver at each iteration step. Let us now list the objectives that are optimized. The matrix and vector Q and c from eq. (1) are weighted sums of the single-objective $(Q_i)_i$ and $(c_i)_i$ corresponding to the objectives listed in the following.

1) Feature tasks: A feature is any vector function f of the robot configuration q:

$$y = f(q) \tag{7}$$

A feature task is defined by a reference task value y^*. We map this reference value to a reference feature acceleration $\ddot{y}^*$ by using a proportional-derivative (PD) control law:

$$\ddot{y}^* = -\lambda_p (y - y^*) - \lambda_D \dot{y} \tag{8}$$

where λ_p and λ_D are feature-specific gains. For the examples presented in the experimental section, we set $\lambda_D = 2\sqrt{\lambda_p}$. Noting J the corresponding jacobian ($J = \frac{\partial f}{\partial q}$), we can express the feature acceleration as a function of the robot configuration derivatives:

$$\ddot{y} = J\ddot{q} + \dot{J}\dot{q} \tag{9}$$

The motion objective for such a feature task is the minimization of the distance between the feature acceleration and the reference feature acceleration:

$$E_f(u) = ||\ddot{y} - \ddot{y}^*||^2 \tag{10}$$

The examples presented in the experimental section use different kind of feature tasks:

- 6D (position, rotation) transformation task on points of the robot, for example for foot transformations;
- 2D parallel task, to keep a vector associated with a robot body parallel to a world frame vector, for example to constrain the chest to stay vertical;
- 2D or 3D CoM position;
- Whole-body configuration task, towards a natural rest pose.

2) Angular-Momentum Task: The total Angular-Momentum (AM) of the robot is a linear function of the configuration first derivative $\dot{q}$. Based on biomechanical observations [23], and following similar strategies in computer animation [12], we regulate the AM L around the robot CoM during locomotion. Let L^* be the reference AM, linear control gives:

$$\dot{L}^* = -\lambda_p (L - L^*) \tag{11}$$

The linear relationship between L and $\dot{q}$ is written:

$$L = J_{AM} \dot{q}, \tag{12}$$

from which follows:

$$\dot{L} = J_{AM} \ddot{q} + \dot{J}_{AM} \dot{q} \tag{13}$$

The motion objective for the angular-momentum task is thus the minimization of:

$$E_{AM}(u) = ||\dot{L} - \dot{L}^*||^2 \tag{14}$$

The examples shown in the experimental section were generated with ($L^* = 0$), which corresponds to damping rotations [24].

3) Joint Torque Minimization: To achieve human-like walk, we add an objective to minimize some actuated joint torques.

$$E_\tau(u) = ||T\tau||^2, \tag{15}$$

where T is a selection matrix. We found that minimizing leg joint torques could lead to instable locomotion patterns, so we only use this to minimize arm or head joint torques. It produces stylistic motions like natural-looking arm or head sway.

4) Task Weights: The weights of the task have been set to account for their different priority in maintaining balance. The task of fixing the position of support feet has a weight of 100, CoM and swinging foot position tasks have a weight of 10, chest orientation task has a weight of 1, and the configuration task towards a rest position has a weight of 0.1. The values of the weights of angular-momentum and joint torque minimization tasks are set by users. This is detailed in the next section.

C. Locomotion State Machine

We have presented the control that takes place at every simulation step. To achieve locomotion, the controller parameters depend on a higher-level finite-state machine. The robot can be in three states: left foot support, right foot support and double support. The tasks that vary depending on the robot state are the foot transformations and the CoM position. When a foot is supporting, foot transformation keeps the foot static. When a foot is swinging, it follows a trajectory computed relatively to the supporting foot, based on a few locomotion parameters (step length, step height).

The CoM x coordinate along the tangent to the robot trajectory is servoed to the middle of the feet. The CoM y coordinate (horizontal, orthogonal to the robot trajectory) follows a $\sin$ function, of period the duration of a complete locomotion cycle. This CoM y trajectory depends on two parameters: its amplitude and a time phase realtively to foot trajectories. We have tried two different possibilities regarding the CoM vertical coordinates z: letting it free or servoing it to a fixed height. The results are not very different.

The locomotion is a little more stable when we fix the CoM height. The experimental section presents results generated with the latter option.

IV. CONTROLLED PARAMETERS AND USER INTERFACE

Having described the general functionning of our controller, let us list the parameters that the user can change in real-time to create new locomotion styles. Fig. 2 shows an example of the graphical interface available to control the locomotion style.

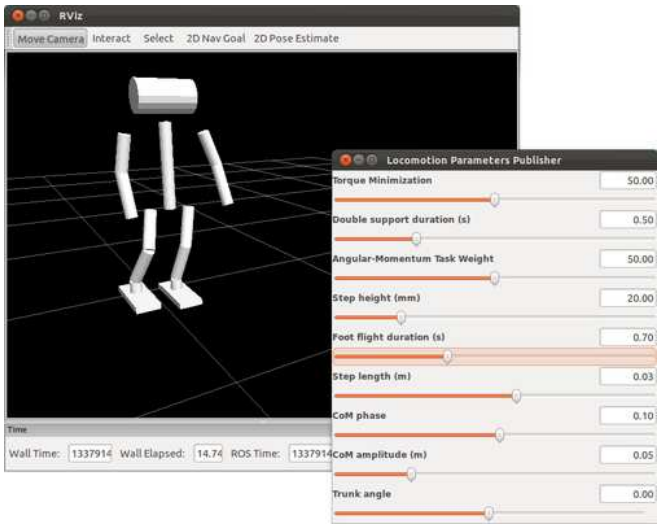

Fig. 2. Screen capture of the user control interface

A. Gait Geometric Parameters

As specified in the previous section, the feature tasks controlling the foot positions and orientations depend on a few geometric parameters. They can be changed interactively by the user:

- Step Length: the difference between the support foot position and the target position for the swinging foot. It can be negative to generate backward walk;
- Step Height: the maximum height of the swinging foot trajectory.

The user can also change offline the whole-body configuration used as a rest pose during the motion. This was used in the presented experiments to change the default position of the head or arms during locomotion.

B. Balance Parameters

The usual way to generate dynamically balanced walk motions for humanoid robots is to plan a Zero-Momentum Point (ZMP) [25] trajectory, that goes from footprint to footprint. Based on this trajectory and a few approximations, it is possible to use preview controllers to generate a corresponding CoM trajectory, see for example [26].

In our work, we found that using preview control allowed less flexibility for the user when creating new locomotion patterns. The ZMP is not explicitely controlled when using our controller. Instead, we only ensure that the dynamic equation of the system is respected at each control step, and that the normal contact forces at the contact points are positive. [18] shows how this condition is equivalent to the classic ZMP criterion. Because our controller does not plan for future ZMP trajectories, there is no guarantee that the robot will always be able to stop and recover static balance. The task of defining a well balanced walk motion is left to the user, but it turned out to be fairly easy with only a few intuitive controls. Following is the list of user-controlled parameters that directly affect balance:

- CoM y trajectory amplitude;
- CoM y trajectory phase, with regards to foot trajectories;
- Duration of double foot support phase;
- Duration of single foot support phase.

The actual balance depends on coupling of these parameters, [27] gives the intuition that the higher the frequence of steps is, the less the amplitude of the CoM trajectory should be. This simple idea was used to define default walk trajectories. Note that the user is free to define unbalanced motions. For example, if the user tries to generate foot trajectories without a corresponding CoM motion, the QP problems will soon be unsolvable and the robot will fall after one or two steps.

C. Stylistic Parameters

Based on relevant computer animation literature, we let users control some parameters that define the style of walk motion. As presented in Section III, the motion objective optimized by the QP solver is a weighted sum of basic task objectives. The tasks consisting in regulating the AM to 0, and the task minimizing joint torques are parameter free. Through the proposed interface, the user is able to set their weight in the global motion objective computation, and thus change the style of motion. The last proposed parameter is the orientation of the chest of the robot. By default it is constrained to stay vertical. The user can change an angular value to make the robot lean forward or backwards. In [13], a user study based on virtual characters shows how chest and head orientations affect the perceived valence of a walking motion. Following is a summary of the user-controlled stylistic parameters:

- Angular-momentum task weight;
- Torque minimization task weight;
- Chest orientation.

V. IMPLEMENTATION DETAILS

The proposed interface has been implemented using ROS middleware [28]. The simulation graphic environment consists of a 3D visualization window, to monitor the robot state, and a simple control window, where sliders positions define locomotion parameters. The QP solver used by our controller is qpOases [29], [30], which implements an online active set strategy. For simulation and control, we use a time step of 5 ms, while total dynamic computation and QP solving requires between 7 and 9 ms. Even if real-time simulation

and control was not achieved, we found that this rate was fast enough to interactively design walking animations.

In simulation, the robot model is controlled by joint torques. However, we did not have access to a physical model of torque-controlled robot. To use the generated locomotion trajectories on a joint angle servo-controlled robot, we saved the generated joint angle trajectories and replayed them on the robot. This is how the results presented in the experimental section were produced.

VI. EXPERIMENTAL RESULTS

The results were obtained on a model of the Aldebaran humanoid robot Nao. Starting from a balanced walking pattern, designing a new one with different style took approximately half an hour, including testing on the physical robot. Sometimes, motions that are balanced in simulation turn out to be infeasible on the real hardware, because of floor and foot surface friction modelization errors. The design of a new locomotion pattern includes testing on the robot. When the real robot was not able to achieve the designed motion, we went back to simulation and tried a more conservative walk.

We present here four different expressive locomotion patterns, that illustrate the use of the stylistic parameters listed above. Readers can refer to the attached video for the complete set of parameters used to design these locomotion controllers. The video presents the simulation interface and physical testing on the robot for all motions.

A. Relaxed Walk

Starting from a balanced forward walk motion, we increase the weights of the angular-momentum control task and the of joint torque minimization task to 50. This automatically produces natural looking arm and head sway, that counterbalances the angular momentum due to CoM motion. It makes the walk look relaxed. Fig. 3 presents the generated relaxed walk.

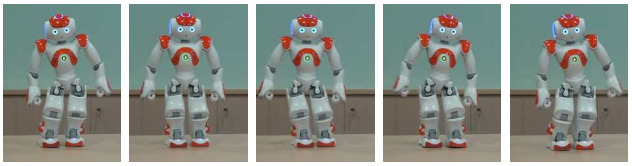

Fig. 3. Relaxed walk motion. Starting from a normal walk, the angular-momentum and torque minimization tasks are given higher weight.

Note that the motion played on the robot does not include feedback control, it is open-loop. Differences between the simulated friction of the contact surface and the real one can lead to errors in the motion execution. In the presented video, the motion is played on a slippery surface, which produces wider body sway than in simulation. The generated motion was stable enough to be executed on this surface.

B. Cautious Walk

For this locomotion pattern, we increase the height and duration of steps. The rest configuration was changed to make the robot lift up its arms and look to the ground in front of him. The resulting motion looks like a cautious walk on an unknown terrain. Fig. 4 shows this cautious walk.

Fig. 4. Cautious walk: the robot makes high and slow steps, while looking at the ground.

C. Proud Walk

[13] shows how chest and head orientations change the perceived valence of a walk motion. Leaning backwards during locomotion is associated with positive valence, and leaning forward with negative valence. We have tested this idea by providing users a way to control the angle between the robot chest and the vertical.

The proud locomotion pattern was generated from a relaxed walk by increasing angular-momentum damping and leaning backwards. Fig. 5 shows this walk pattern.

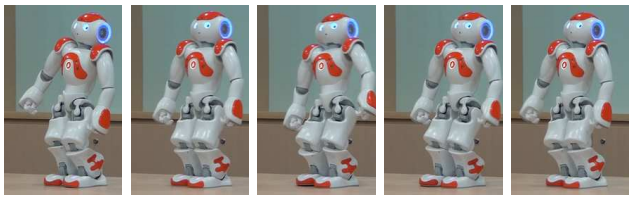

Fig. 5. Proud walk: the robot leans backwards, minimizes joint torques and controls its angular-momentum.

D. Sad Walk

The sad walk was generated from a relaxed walk motion by making slower and lower steps. The robot is leaning forward, and the angular-momentum control is reduced. Fig. 6 shows the sad walk pattern.

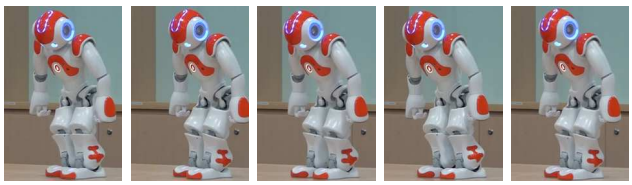

Fig. 6. Sad walk: the robot leans forward, makes small steps and minimizes joint torques.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have presented an interactive dynamic controller for humanoid robots, used to generate stylistic locomotion patterns. Through a few graphical controls, users can generate locomotion trajectories that vary in their dynamic properties and style. We found that expressing the controlled motion features at a dynamic level allowed the generation of natural looking walk motions. Our controller

does not include balance preview control, so the user is free to define unbalanced motions. We found that this allowed more flexibility in the exploration of the range of possible motion, at the price of a short learning phase for the user. We believe that providing motion artists with flexible and intuitive software is necessary to the development and use of social humanoid robots.

In the future, we plan to conduct deeper research on the choice of controlled features. We would like to automatically choose them, based on motion data set analysis. This could be applied to other types of motion, not just locomotion. Future user studies will evaluate the relevance of the controlled features and the ease to use the interface. We also plan to evaluate the perceived expressiveness of the motions generated with our interface. Finally, we plan to use our interface to generate expressive motions on more complex humanoid models.

ACKNOLEDGEMENTS

This research, which is carried out at BeingThere Centre, is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

REFERENCES

[1] R. Picard, "Affective computing: challenges," *International Journal of Human-Computer Studies*, vol. 59, no. 1, pp. 55–64, 2003.

[2] M. Pasch and R. Poppe, "Person or puppet? the role of stimulus realism in attributing emotion to static body postures," *Affective Computing and Intelligent Interaction*, pp. 83–94, 2007.

[3] S. Nakaoka, S. Kajita, and K. Yokoi, "Intuitive and flexible user interface for creating whole body motions of biped humanoid robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2010, pp. 1675–1682.

[4] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, "Adapting human motion for the control of a humanoid robot," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1390–1397.

[5] K. Yamane and Y. Nakamura, "Dynamics filter-concept and implementation of online motion generator for human figures," *Robotics and Automation, IEEE Transactions on*, vol. 19, no. 3, pp. 421–432, 2003.

[6] P. Glardon, R. Boulic, and D. Thalmann, "Pca-based walking engine using motion capture data," in *Proceedings of the Computer Graphics International*. IEEE, 2004, pp. 292–298.

[7] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," in *ACM Transactions on Graphics (TOG)*, vol. 21, no. 3. ACM, 2002, pp. 473–482.

[8] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[9] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[10] L. Sentis, "Synthesis and control of whole-body behaviors in humanoid systems," Ph.D. dissertation, STANFORD UNIVERSITY, 2008.

[11] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.

[12] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," *ACM Transactions on Graphics (TOG)*, vol. 29, no. 4, p. 131, 2010.

[13] M. Inderbitzin, A. Valjamae, J. Calvo, P. Verschure, and U. Bernardet, "Expression of emotional states during locomotion based on canonical parameters," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*. IEEE, 2011, pp. 809–814.

[14] J. Hodgins, W. Wooten, D. Brogan, and J. O'Brien, "Animating human athletics," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM, 1995, pp. 71–78.

[15] P. Faloutsos, M. Van De Panne, and D. Terzopoulos, "Composable controllers for physics-based character animation," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 251–260.

[16] J. Laszlo, M. van de Panne, and E. Fiume, "Interactive control for physically-based animation," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 201–208.

[17] K. Yin, K. Loken, and M. van de Panne, "Simbicon: Simple biped locomotion control," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 105, 2007.

[18] L. Saab, O. Ramos, N. Mansard, P. Soueres, and J. Fourquet, "Generic dynamic motion generation with multiple unilateral constraints," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2011, pp. 4127–4133.

[19] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 4414–4419.

[20] O. Ramos, L. Saab, S. Hak, and N. Mansard, "Dynamic motion capture and edition using a stack of tasks," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*. IEEE, 2011, pp. 224–230.

[21] R. Featherstone, *Robot dynamics algorithms*. Kluwer Academic Publishers, 1987.

[22] Y. Abe, M. Da Silva, and J. Popović, "Multiobjective control with frictional contacts," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 2007, pp. 249–258.

[23] H. Herr and M. Popovic, "Angular momentum in human walking," *Journal of Experimental Biology*, vol. 211, no. 4, pp. 467–481, 2008.

[24] M. Popovic, A. Hofmann, and H. Herr, "Angular momentum regulation during human walking: biomechanics and control," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 3. IEEE, 2004, pp. 2405–2411.

[25] M. Vukobratovic and B. Borovac, "Zero-moment point-thirty five years of its life," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 157–173, 2004.

[26] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 2. IEEE, 2003, pp. 1620–1626.

[27] S. Dalibard, A. Khoury, F. Lamiraux, M. Taix, and J. Laumond, "Small-space controllability of a walking humanoid robot," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*. IEEE, 2011, pp. 739–744.

[28] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009.

[29] H. Ferreau, H. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit mpc," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.

[30] A. Potschka, C. Kirches, H. Bock, and J. Schlöder, "Reliable solution of convex quadratic programs with parametric active set methods," Interdisciplinary Center for Scientific Computing, Heidelberg University, Im Neuenheimer Feld 368, 69120 Heidelberg, GERMANY, Tech. Rep., November 2010.