A Simplified Contact Model for Treating the Balance of Biped Virtual Characters

Danilo Borges da Silva*
UFC

Abstract

In this work, we present a physically-based model for maintaining the equilibrium of biped characters. We use Proportional-Derivative (PD) Controllers to mimic the characteristics of angular joints, and deal with the equilibrium through a method that uses the transpose of the Jacobian matrix that relates the angular and linear momentum at the Center of Mass (COM) of the character's body to the corresponding quantities at all its members, which are treated as end effectors. Namely, the balance control involves all links, as a whole, and not only the lower body. Also, we propose a simple contact model for treating the interaction between the feet and the ground that confers great stability to the character and makes it easier for the controller and for the starting of the simulation. It also makes it convenient for the animator to adjust the model so as to consider a trade-off between more physical realism and more stability. The system does not use optimization and is simple to implement. Unlike existing methods for control based on reference motions, our framework does not require preprocessing of the reference motion, nor does it rely on inverse dynamics or optimization methods. The robustness of the controller is demonstrated through a series of tests that show that the character is able to adapt its posture and maintain equilibrium either under the influence of external actions from the environment or when it follows unaltered reference motions.

Keywords: 3D interaction, virtual humans and avatars, physics-based animation, controllers

1 Introduction

In the field of physics-based control of human motion, a simulated character usually consists of a structure of articulated rigid bodies (links) with internal angular actuators located at the joints. The feet are usually represented in a very simplified way, through boxes (parallelepipeds), and, for that reason, the interaction with the ground is very discontinuous [Lee et al. 2010; Jain and Liu 2011]. That type of model (articulated rigid links and simplified rigid feet plus angular actuators), however, is capable of capturing the fundamental aspects of the human musculoskeletal system. Therefore, researchers were able to develop successfully a number of controllers for performing multiple tasks involving balance [Geijtenbeek et al. 2013]. Despite the remarkable achievements, the simplified model still imposes a constraint that prevents the virtual human from exhibiting the same abilities of a real human. Thus, more robust models have been proposed recently [Jain and Liu 2011; Wang et al. 2012].

*e-mail:danilobs@lia.ufc.br

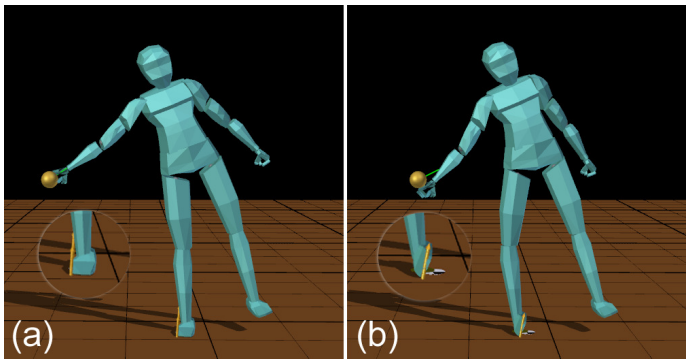

Figure 1: *The stance foot contact model. (a) The proposed simplified contact model increases the stability between the foot and the ground, allowing the stance foot remains steady, even when the used geometry in the simulation is not appropriate. (b) Without the proposed model, the internal torque applied to the foot is not compensated. In this case, more complex controllers, involving optimization [Macchietto et al. 2009; Geijtenbeek et al. 2012], would be required.*

The balance of a virtual character at a given instant depends upon the complex interrelation between the set of external forces and the set of internal forces produced by the actuators, which are located at the joints, and that reconfigure each individual rigid link of the articulated structure, altering the overall posture's configuration. The dynamic reconfiguration of the posture modifies the external contact forces continuously. Thus, if the only parts of the body in contact with the ground are the feet, the way each individual foot is modeled will have a strong influence on the overall distribution of the external contact forces, and, therefore, on the balance of the character. Because a real foot is structurally complex and flexible, it can generate an infinite number of external contact forces redistributions when trying to maintain the global equilibrium. The feet models used for simulations are much less complex and, therefore, cannot reproduce the same redistribution of contact forces obtained with real feet. Jain and Liu [2011] have shown that, when the complexity of the foot model increases, an existing controller becomes more robust because it is able to represent the distribution of contact forces better.

In this work, we argue that the use of a stable contact model facilitates the design of robust controllers, without the need of a sophisticated model for the foot of the virtual human. We believe that modeling a realistic foot that offers the same range of redistribution of contact forces that a real foot offers is important. However, that still requires a great research effort [Jain and Liu 2011]. Therefore, in a simulation, we propose to uncouple the foot's model from the foot's physical interaction with the ground.

Regarding two extreme situations, in which the friction between the foot and the ground is either infinite or zero (or buried foot vs. wooden leg), is easy to see that the more firmly the foot is on the ground, the more control over the global equilibrium the character will have. From the point of view of a physical simulation, keeping the foot firmly on the ground implies a null resultant torque. Therefore, since an instantaneous equilibrium condition implies not only

global equilibrium, but also the equilibrium of each individual link of the character's structure, the forces acting on the foot must be also in equilibrium. So, what we seek is to maintain the equilibrium between the ground forces acting on the foot with the internal forces transmitted by the actuators connected to the foot. This is a control problem because those forces are constantly being updated. Figure 1 illustrates what happens when the torque applied to the foot by the actuators is not in equilibrium with that produced by the Ground Reaction Forces (GRFs).

What is proposed in this work is to produce, artificially, the torques that would be provided by the ground forces generated by the accommodation of a complex foot with the ground. Those artificial torques are generated independently of the foot model used, and, therefore, simplify the control of the foot vs. ground interaction. However, although the artificial torques are not compatible with the foot's geometry indeed used in the simulation, they have plausible values, and still can be considered physically correct, validating the generated motion. The main advantages of this approach are its simplicity and its independence from a sophisticated foot model.

It is obvious that, if no bounds are imposed to the artificially generated torques, the physical realism may be lost and the character would be endowed with superpowers. However, sacrificing the motion's physical correctness in order to gain stability may be desirable in some situations. The proposed technique explores that flexibility of controlling stability in order to provide a robust tool that gives more freedom to the animator.

The Jacobian matrix of the mapping between two coordinate systems has been used as an abstraction layer to simplify the control of balance [Coros et al. 2010; Geijtenbeek and Pronost 2012; Geijtenbeek et al. 2012]. That abstraction allows the center of mass (COM) of the character to be controlled directly through the application of virtual forces and torques to the COM. Those virtual forces and torques are, in fact, converted to equivalent internal torques that are applied to the joints by the actuators. Most of the works in the literature do not consider the influence of the upper part of the character in the construction of the Jacobian. In this paper, however, we present a construction based on the total momentum of the character at its COM, allowing the control of balance to involve the character as a whole, and not only its lower limbs.

2 Related Work

In this section, we present the most relevant works and discuss some of their differences with respect to the current work. Two main aspects are considered in the discussion: the modeling of the foot and the way the Jacobian matrices are used for stability control.

Some authors argue that, in order to achieve greater stability of locomotion, the model of the character's foot should reproduce as close as possible the behavior of a real human foot. Wang et al. [2009] increased the number of joints to model a more flexible foot that essentially consists of a greater number of rigid links. Jain and Liu [2011] modeled the foot as a deformable object in order to test the hypothesis that "ignoring the effect of deformable bodies at the site of contact negatively affects the control algorithms, leading to less robust and unnatural character motions." Geijtenbeek et al. [2012], in turn, seem to have needed to use a foot with larger proportions, in order to increase the contact area with the ground and, thus, facilitate the achievement of successful results.

Considering the works that use Jacobian matrices for stability control, Coros et al. [2010] incorporate a form of Jacobian transpose [Sunada et al. 1994] to motion control resulting in versatile and generic actions for bipedal locomotion. That control method is based on the control system of Pratt et al. [2001], which use virtual actuators. That same system is used as a basis to provide balance to biped characters in [Geijtenbeek et al. 2012; Zordan and Hodgins 2002]. Regarding the construction of the Jacobian, the global DOFs of the character (not actuated) should not be restricted to those related to the character's global position only. Thus, unlike the work of Coros et al. [2010], it is interesting that the character's global orientation is also considered in the construction of the Jacobian, allowing the application, at the COM, of virtual forces as well as of virtual torques. This paper considers global orientations and positions in a unified way. Geijtenbeek et al. [2012] also use global orientations and positions, but not in a unified manner.

While some authors [Abe et al. 2007; Macchietto et al. 2009; Wang et al. 2009; Geijtenbeek et al. 2012] need to use optimization, even if offline, in order for their equilibrium strategies to work well, our contact model provides a stable and easily adjustable alternative for the treatment of balance without optimization. Also, while Coros et al. [2010] use pose control graphs [Yin et al. 2007] to design the style of the motion generated by the controller, the model proposed in this work allows that unaltered captured motions be used for this purpose. The work of Geijtenbeek and his co-workers [Geijtenbeek et al. 2012] also presents that functionality, but it requires optimization.

3 Overview

As illustrated in Figure 2, the structure of the proposed controller has three main components: PD Controllers, Balance Control and Simplified Contact Model of the feet with the ground.

The PD controllers guarantee that the poses provided either by the user or by motion capture are achieved. In order to do that, they apply torques to the joints according to a hierarchy that considers the pelvis as its root. However, PD controllers do not control the global DOFs directly. In fact, those DOFs are controlled by means of virtual forces and torques that are applied to the character's COM.

The balance control component is responsible for computing those virtual forces and torques and converting them into equivalent internal torques, through the transpose of the Jacobian matrix. The Jacobian matrix is assembled according to a hierarchy that considers the support foot as its root. Note that, in some situations, both feet may be considered simultaneously as the support.

Finally, the torque (or a fraction of it) applied to the support foot is artificially compensated in order to increase the stability of interaction with the ground and thereby facilitate the control of balance. Each one of the controller's components is discussed in more detail in the following sections.

4 PD Controllers

The implementation of the PD controllers is based on [Nunes 2012], in which quaternions are used to represent the 3D orientations. Considering only spherical joints, each torque applied to a joint, j, is given by:

$$^{j}\tau_{pd} = k_s(q_d q_a^{-1}) + k_d(\omega_d - \omega_a), \quad (1)$$

where q_a and q_d are, respectively, the current and the desired quaternions of the joint, ω_a and ω_d are the current and desired angular velocities of the joint, and k_s and k_d are user-defined constants. The expression $q_d q_a^{-1}$ corresponds to the three-dimensional version of the following two-dimensional angular difference $(\theta_d - \theta_a)$. Note that the resulting quaternion still needs to be converted to the axis and angle representation. The desired information can be obtained directly from captured motions, without any adjustment. In

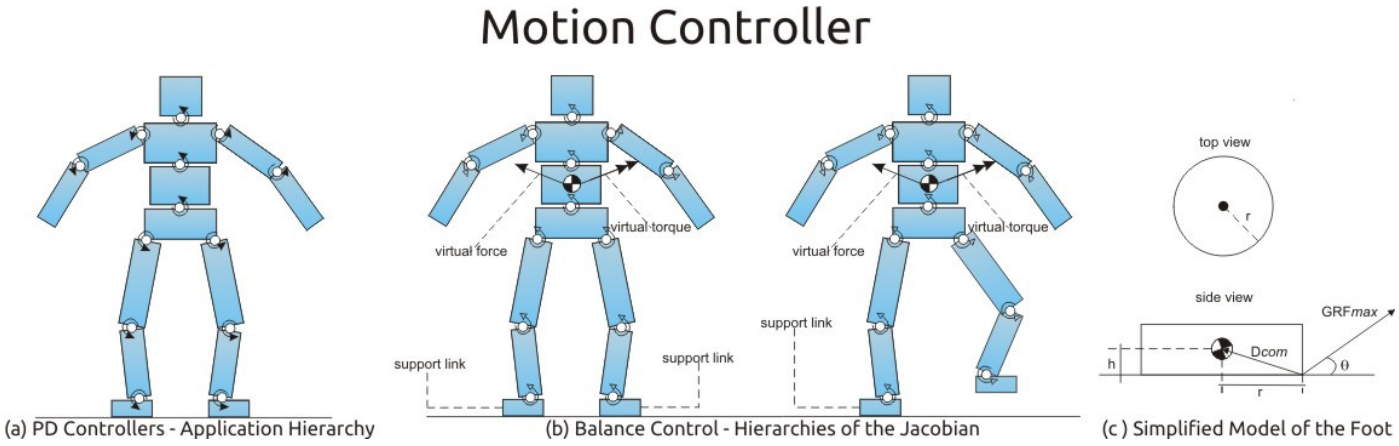

Figure 2: *Motion controller overview. (a) PD controllers are used to mimic the angular characteristics extracted from reference motions or specified by the animator. (b) The Balance Control of the character uses a Jacobian transpose to compute the internal torques to be applied at its joints, according to the hierarchy defined by the link that is in contact on the ground, from the virtual force and virtual torque applied at its center of mass. (c) The simplified contact gives to the character greater stability, compensating part or the whole impact of its foot against the ground.*

contrast, some works require large adjustments [Yin et al. 2007]. According to a pre-established hierarchy, applying a torque τ to a joint corresponds to applying the same torque τ to its child link and the opposite torque -τ to its parent link.

5 Balance Control

As already stated, the balance control is required because the character's global DOFs do not have direct actuators. Shortly, a Jacobian matrix is used to convert virtual forces and torques, to be applied to the COM, in equivalent internal joint torques. This section is divided into three main parts: construction of the Jacobian matrix, calculation of the virtual force and calculation of the virtual torque.

5.1 Jacobian's Construction

The Jacobian matrix describes the linear relationship between the velocity of a specific point (end effector) and the velocities of the joints which influence that point, according to a predefined hierarchy. Note the Jacobian involves only a sub-chain of articulated links, starting from a fixed base (e.g. stance foot). To treat the balance, the COM of the character is usually chosen as the end effector. Using the *spatial vector notation* (Appendix .1), we want to determine the Jacobian J_{com}:

$$^{com}\phi_{com} = J_{com}\ ^{J}\Phi_{J} \tag{2}$$

where $^{b}\phi_{a}$ represents the *spatial velocity* of a with respect to the frame b (i.e., in the coordinates of b), and $^{J}\Phi_{J}$ corresponds to the vector formed by all joints' *spatial velocities* together, in each joint's local coordinates. Considering the character has n joints in total, $^{J}\Phi_{J} = \left(^{j_0}\phi_{j_0}^{T}\ ^{j_1}\phi_{j_1}^{T}\ ...\ ^{j_{n-1}}\phi_{j_{n-1}}^{T}\right)^{T}$ has dimension $6n\times 1$.

However, considering the COM as a simple end effector means assuming that only some joints influence it, what clearly is not true. In this work, all links are considered as end effectors, each having an exclusive Jacobian. The velocities of all links, as well as their individual Jacobians, are then proportionally combined, according to their masses, in order to calculate the character's total momentum, relative to its COM:

$$^{com}\begin{pmatrix} L \\ P \end{pmatrix}_{c} = {}^{com}\begin{pmatrix} \mathcal{I}\cdot\omega \\ m\cdot v \end{pmatrix}_{c} = (^{com}_{c}M)(^{com}\phi_{com}), \tag{3}$$

where L corresponds to angular momentum, P, to linear momentum, $\mathcal{I}$, to inertia matrix, ω, to angular velocity, m, to mass, and v, to linear velocity. The c index encompasses the total information of the character. $^{com}_{c}M$ is the total *spatial mass* of the character, in the coordinates of its COM.

The total momentum can also be obtained by the sum involving all links:

$$^{com}\begin{pmatrix} L \\ P \end{pmatrix}_{c} = \sum_{l}(^{com}_{l}M)(^{com}\phi_{l}) = \sum_{l}(^{l}_{com}Ad)^{T}(^{l}_{l}M)(^{l}\phi_{l}), \tag{4}$$

where the adjoint matrix $(^{l}_{com}Ad)^{T}$ transforms the spatial momentum of each link l, given in local coordinates, in the coordinates of the COM. Note that spatial momentum, as well as *spatial force*, is transformed with the inverse transpose of the adjoint matrix.

As mentioned, each link l has an exclusive Jacobian, J_l, which relates your *spatial velocity*, $^{l}\phi_{l}$, with the vector $^{J}\Phi_{J}$. Note that, according to a particular hierarchy, $^{l}\phi_{l}$ can be obtained by the sum of the *spatial velocities* of all joints influencing the link l:

$$^{l}\phi_{l} = \sum_{j}{}^{l}\phi_{j} = \sum_{j}{}^{l}_{j}Ad\ ^{j}\phi_{j} = J_{l}\ ^{J}\Phi_{J}, \tag{5}$$

where the sum in j includes all joints that influence the link l. Note that, in order to isolate the whole vector $^{J}\Phi_{J}$ at right, that sum in j could be replaced by the multiplication of J_l and $^{J}\Phi_{J}$. Therefore, J_l corresponds to a matrix, with dimension $6\times 6n$, containing those adjoint matrices horizontally arranged at the locations corresponding to their respective joints, included in the sum. Zero matrices with dimension 6×6 are placed at the locations corresponding to the joints that do not influence the link l. For clarity, consider hypothetically that the character has only 6 joints. Also consider that, according to the adopted hierarchy, a link l is influenced by the joints j_1, j_2 and j_4. The individual Jacobian of that link l is defined as follows:

$$^{l}\phi_{l} = J_{l}\ ^{J}\Phi_{J} = \begin{bmatrix} 0 & {}^{l}_{j_1}Ad & {}^{l}_{j_2}Ad & 0 & {}^{l}_{j_4}Ad & 0 \end{bmatrix}\ ^{J}\Phi_{J}. \tag{6}$$

In order to facilitate the implementation, a given hierarchy may be represented by a table relating all joints (lines) to all links (columns) of the character. Each cell of that table is filled with 0 (zero) or 1 (one), according to the hierarchy. Filling a cell with 1 means that the joint corresponding to that line influences the link corresponding to that column. Otherwise, the cell should be filled with 0. At each step of the simulation, both the hierarchy of the character and the Jacobians are updated according to the contacts between the feet and the ground.

Finally, combining the Equations 3, 4 and 5, and comparing with the Equation 2, we figure out the final expression of the Jacobian J_{com}, which involves all joints of the character, and not only the lower limbs:

$$J_{com} = ({}^{com}_{c}M)^{-1} \sum_{l} ({}_{com}{}^{l}Ad)^{T} ({}^{l}_{l}M) J_l, \tag{7}$$

where:

$${}^{com}_{c}M = \sum_{l} ({}_{com}{}^{l}Ad)^{T} ({}^{l}_{l}M) ({}_{com}{}^{l}Ad). \tag{8}$$

The operation $({}_{com}{}^{l}Ad)^{T} ({}^{l}_{l}M) ({}_{com}{}^{l}Ad)$ transforms the *spatial mass* matrix of each link l, given in local coordinates, in the coordinates of the COM [Nunes 2012].

The internal *spatial forces*, to be applied to the joints of the character, are obtained by the transpose of the Jacobian J_{com}, according to the virtual *spatial force*, composed by the concatenation of the virtual torque and the virtual force (Subsections 5.3 and 5.2):

$${}^{J}W_J = (J_{com})^{T} \, {}^{com}w_{com}, \tag{9}$$

where ${}^{b}w_a$ is the *spatial force* of a in the coordinates of b, and ${}^{J}W_J$ corresponds to the vector formed by all joints' *spatial forces* together, in each joint's local coordinates. Similar to ${}^{J}\Phi_J$ (Equation 2), ${}^{J}W_J = \left({}^{j_0}w_{j_0}^{T} \; {}^{j_1}w_{j_1}^{T} \; ... \; {}^{j_{n-1}}w_{j_{n-1}}^{T}\right)^{T}$ also has dimension $6n \times 1$.

Note that, as only ball (spherical) joints are used, only torques (angular part) are required from those obtained internal *spatial forces*. The coordinates corresponding to the forces (linear part) are simply ignored. Thus, the torque to be applied to the joint j of the character, at each instant of the simulation, is computed as:

$${}^{j}\tau_{total} = {}^{j}\tau_{pd} + {}^{j}\tau_{bal}, \tag{10}$$

where ${}^{j}\tau_{bal}$ corresponds to the angular part (torque) of the internal *spatial force* obtained for the joint j, using the Equation 9.

5.2 Virtual Force

The expression of the virtual force has three main terms.

The first term is responsible for maintaining the COM of the character over its support region, which is generally simplified and replaced by a single point, here called the support point, p_{sup}. A PD controller is used to correct the position and velocity of the COM. In the general case, we have:

$$f_{control} = k_{fs} \left(p_{com_d} - p_{com_a}\right) + k_{fd} \left(v_{com_d} - v_{com_a}\right), \tag{11}$$

where p_{com_d} and p_{com_a} are the desired and current positions of the COM, v_{com_d} and v_{com_a} are the desired and current linear velocities of the COM, and k_{fs} and k_{fd} are user-defined constants. Note that keeping the COM over the support point does not imply any vertical control. Therefore, only the horizontal projections of that information should be used.

If no reference motion is used, all information is obtained from the simulation. In addition, the desired position and the desired velocity of the COM respectively correspond to the support point ($p_{com_d} = p_{sup}$) and zero ($v_{com_d} = 0$). Otherwise, when a captured motion is used as a reference, the position and velocity of the COM at the simulation are compared to the position and velocity of the COM at the captured motion:

$$f_{control} = k_{fs} \left({}^{mo}\hat{p}_{com} - {}^{sim}\hat{p}_{com}\right) + k_{fd} \left({}^{mo}v_{com} - {}^{sim}v_{com}\right), \tag{12}$$

where ${}^{mo}\hat{p}_{com}$ and ${}^{sim}\hat{p}_{com}$ are the relative positions of the COM, at the captured motion and at the simulation, respectively.

Note that considering absolute positions would mean tying the global position of the simulated character to the captured motion. Therefore, those positions should be calculated relative to the respective characters, the used to reproduce the captured motion and the simulated one. The support points of those characters are used as reference points to calculate those relative positions: $\hat{p}_{com} = p_{com} - p_{sup}$. Thus, the used controller supports horizontal translations of the simulated character.

Also note that the support point's choice may vary. One option would be to use the center of the convex hull, but it would require unnecessary calculations for the purpose of this work. Simplest choices, based on the positions of the ankles or the COMs of the feet, are usually enough [Abe et al. 2007] [Geijtenbeek et al. 2012]. In this work, the support point's choice depends on the contact situation between the character and the ground. If only one foot is in contact with the floor, p_{sup} is calculated as the COM of that stance foot, horizontally projected. If both feet are in contact with the floor, p_{sup} is calculated as the midpoint between the COMs of the two feet, also horizontally projected.

The second term is responsible for compensating gravity. For this purpose, a constant vertical force should be applied to the COM of the character, contrary to its weight. The Jacobian used in this work allows that force to be easily considered as part of the virtual force:

$$f_g = -\sum_{l} m_l g, \tag{13}$$

where the sum in l includes all links of the character, m_l is the mass of each link l and g is the gravity acceleration.

The third and last term is responsible for correcting the character's linear momentum, relative to its COM:

$$f_P = k_P({}^{mo}P - {}^{sim}\,P), \;\; P = \sum_{l} m_l v_l, \tag{14}$$

where k_P is a user-defined constant and v_l is the velocity of each link l. The velocities of the links at the captured motion, ${}^{mo}v_l$, are estimated using finite differences. In the absence of reference motions, ${}^{mo}P = 0$ is considered.

The final expression for the virtual force is defined as:

$$f_{virtual} = f_{control} + f_g + f_P. \tag{15}$$

5.3 Virtual Torque

The expression of the virtual torque has two main terms.

The first term is responsible for controlling the global orientation of the character, which is performed by controlling a specific link:

$${}^{l}\tau_{control} = k_{ts}({}^{l}q_d \, {}^{l}q_a{}^{-1}) + k_{td}({}^{l}\omega_d - {}^{l}\omega_a), \tag{16}$$

where ${}^{l}q_a$ and ${}^{l}q_d$ are the current and the desired quaternions of the link l, in global coordinates, ${}^{l}\omega_a$ and ${}^{l}\omega_d$ are the current and desired angular velocities of the link l, and k_{ts} and k_{td} are user-defined constants. The desired information can be obtained from reference motions or supplied by the user. Instead of applying external torques directly to the links [Wrotek 2006], ${}^{l}\tau_{control}$ is considered as part of the virtual torque, which will be converted by the transposed Jacobian into internal torques at the joints. For the tests, the chosen link was the chest, since it was considered the more stable link, with less changes of orientation, during the simulation.

The other term is responsible for correcting the character's angular momentum, relative to its COM:

$$\tau_L = k_L({}^{mo}L - {}^{sim}L), \tag{17}$$

where k_L is a user-defined constant and

$$L = \sum_l \left(L_l + (p_l - p_{com}) \times (m_l (\omega_l - \omega_{com}))\right), \tag{18}$$

where the sum in l include all links of the character, $L_l = \mathcal{I}_l\omega_l$ is the angular momentum of each link l (where $\mathcal{I}_l$ and ω_l are its inertia and its angular velocity, respectively), p_l is the COM position of each link l, and p_{com} is the COM position of the character. The angular velocities of the links at the captured motion, ${}^{mo}\omega_l$, are also estimated using finite differences. In the absence of reference motions, ${}^{mo}L = 0$ is considered.

The final expression for the virtual torque is defined as:

$$\tau_{virtual} = {}^{l}\tau_{control} + \tau_L. \tag{19}$$

6 Simplified Contact Model

This paper proposes a simplified contact model of the foot using a parametric geometry, which allows to model it in a generic way, according to the user's desire. Based on *Coulomb's model of friction* [Popov 2010], the parameters influence the calculation of the maximum torque that can be compensated in the foot, allowing the user to balance stability and physical correctness. Thus, it is up to the user to choose different variations between fairly stable but physically incorrect controller and an unstable but physically correct controller (Figure 3).

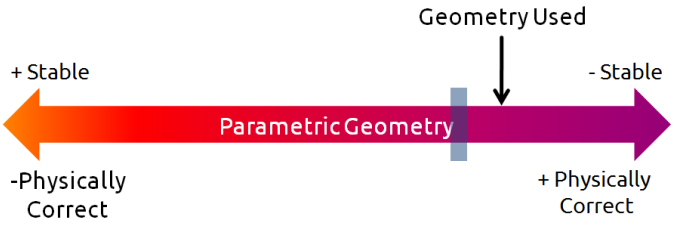

Figure 3: *The proposed parametric geometry of the foot allows the user to balance stability and physical correctness of the simulation.*

It is important to note that there may happen some cases more stable than those granted by the foot geometry indeed used in the simulation (i.e. the blue bar on the left side of the used geometry, as in Figure 3), where the generated motion can still be considered physically correct. In these situations, only the used geometry is not suitable for the desired motion. However, if the used geometry is improved in a realistic manner, in order to increase stability, the previously acquired motion would be validated. Therefore, we can say that the physical correctness of the motion is, in fact, independent of the used geometry. Note also that, in the more unstable case, where no torque is artificially compensated (i.e. the blue bar on the rightmost side), the torque applied to the foot can still be compensated, regardless of the parametric geometry, by the interaction between the geometry of the foot indeed used in the simulation and the ground.

Before explaining in more details the used parametric geometry, it is important to realize that the idea of decoupling the geometry of the foot used in the simulation from its interaction with the ground introduces an abstract representation of this interaction. The concrete structure of this representation, which defines the meaning of the parameters, is not unique. The only requirement is that there should be consistency in relation to the compensation of the torque applied to the foot. After all, the defined structure and the choice of the parameters must determine how much torque applied to the foot should be compensated. Furthermore, it is desired that the manipulation of these parameters be simple. Therefore, although it is necessary to define a specific structure, the contribution of using a detached abstract representation goes beyond the limits imposed by a chosen structure in particular. The structure chosen in this work is described below.

For simplicity, some restrictions on the used structure are defined. First, the geometry chosen to represent the foot has a cylindrical shape, as illustrated in Figure 2c (top and side views). Assuming the illustrated form, one can predict the extreme situation where the maximum torque in the foot would be offset, namely, a resultant ground reaction force being applied to one end of the contact of the cylinder with the ground. Due to the symmetry of the cylinder, without loss of generality, it can be assumed that this resultant force is applied to the rightmost end of the corresponding circle, which represents the contact area between the cylinder and the ground. Based on *Coulomb's model of friction* positioned in this extreme contact point, it is also assumed that the resultant force capable of compensating the maximum torque on the foot is in accordance with the scheme illustrated in Figure 2c, in which there is a vector, $GRFmax$, located in the limit of the cone defined by the angle θ. Whereas the foot has uniform density, it is still necessary to define the dimensions of the cylinder and a maximum length for the vector $GRFmax$. Thus, one can calculate the maximum compensatable torque, τ_{comp}, as:

$$\tau_{comp} = D_{com} \times GRF_{max} = \begin{pmatrix} -r \\ h \\ 0 \end{pmatrix} \times \begin{pmatrix} m\cos\theta \\ m\sin\theta \\ 0 \end{pmatrix}, \tag{20}$$

where h is the distance from the ground to COM of the foot, r is the foot's radius, m is the length of the vector $GRFmax$ and θ is the angle of the friction cone. It is important to emphasize that these four parameters are user defined. And it is through the choice of values for these parameters that the user is capable of, abstractly, sliding the bar illustrated in Figure 3.

Considering all the mentioned assumptions, it is possible to verify a limitation on the chosen structure: τ_{comp} is always horizontal and, thus, is not able to compensate for any vertical component of torque applied to the foot. Again, for simplicity, it is assumed that the vertical component is always completely compensated, in an artificial way. Furthermore, again due to the symmetry of the cylinder, only the length of the maximum compensable torque is necessary in order to determine how much of the torque applied to the foot must be compensated. For any direction of the torque applied to the foot, the extreme point of contact and direction of the vector $GRFmax$ can be conveniently adapted, getting always the same scheme of the Figure 2c.

Consider the following ratio, corresponding to the percentage of the

torque applied to the foot which will be compensated:

$$comp = \frac{\|\tau_{comp}\|}{\|^{foot}\tau_{bal}\perp\|}, \tag{21}$$

where $^{foot}\tau_{bal}\perp$ is the horizontal projection of the torque applied in the stance foot, extracted from the balance control (Equation 10). Note that $^{foot}\tau_{bal} = -^{ankle}\tau_{bal}$ and $\|\tau_{comp}\| = |h\ m\ \cos\theta + r\ m\ \sin\theta|$. Once the value of $comp$ is truncated if it is greater than 1, it will always be in the range [0, 1]. In Figure 3, increasing the value of $\|\tau_{comp}\|$ (and, consequently, of $comp$) corresponds to moving the slider to the left. Therefore, the total torque applied to the support foot of the simulated character is defined according to this ratio of compensation:

$$^{foot}\tau_{total} = ^{foot}\tau_{pd} + ^{foot}\tau_{bal}\perp(1 - comp). \tag{22}$$

When this percentage ($comp$) of compensation is below a certain tolerance value chosen by the user, the balance control can be turned off, allowing the character to fall more naturally. When turning it off, increasing the damping in PD controllers also helps to get a more natural fall.

6.1 Sensor de Contato

Neste trabalho são utilizadas duas metodologias para determinar qual o pé do personagem simulado que é considerado pé de apoio para os cálculos da matriz Jacobiana. Uma é ativada quando o personagem não segue um movimento capturado e a outra é utilizada quando o mesmo é guiado para seguir um movimento capturado. Sendo assim, utilizamos dois tipos de sensores que funcionam de acordo com a metodologia utilizada. Para a primeira metodologia, o sensor leva em consideração a posição dos pés, suas GRFs e a projeção do COM do personagem no solo e no segundo caso considera-se o pé que se presume estar em contato com o solo do movimento capturado e as GRFs do personagem simulado. Em ambos os casos, a metodologia é realizada em cada instante da simulação.

Quando o personagem não segue um movimento capturado ele independe de informações externas para determinar o contato com o solo, sendo assim necessárias apenas informações que o próprio ambiente físico pode promover. Neste caso, as GRFs são ideais para se determinar o contato do pé do personagem com o solo, porém não é suficiente para determinar se o pé em contato será o pé de apoio (para fins de equilíbrio). Sendo assim utilizamos a seguinte metodologia para determinar o pé de apoio do personagem, de acordo com a Figura 4. Quando a projeção do COM encontra-se em um dos círculos de influência e existe GRFs determinamos este pé como sendo um pé de apoio, ou se o COM não se encontrar em nenhuma das zonas de influência dos pés.

No entanto faz-se necessária uma análise da mudança dos pés de apoio, de dois pés para um pé e de um pé para um dois pés. O primeiro caso é simples, basta verificar se o COM está projetado em uma única zona de influência para desligar o outro pé como pé de apoio, no entanto o segundo caso para deixar a transição mais natural realizamos uma alteração na força de controle (responsável por levar o COM para o ponto de equilíbrio desejado). Observa-se a Figura 5, no primeiro caso temos que quando os vetores estão na mesma direção utilizamos a força de controle já calculada anteriormente (Equação 11), caso contrário faremos uma decomposição para determinar a força de controle de modo que faça com que o COM se estabilize em direção ao pé que está no ar de acordo com a seguinte equação:

$$f_{control} = f_{control} - d_u(f_{control} \cdot d_u) \tag{23}$$

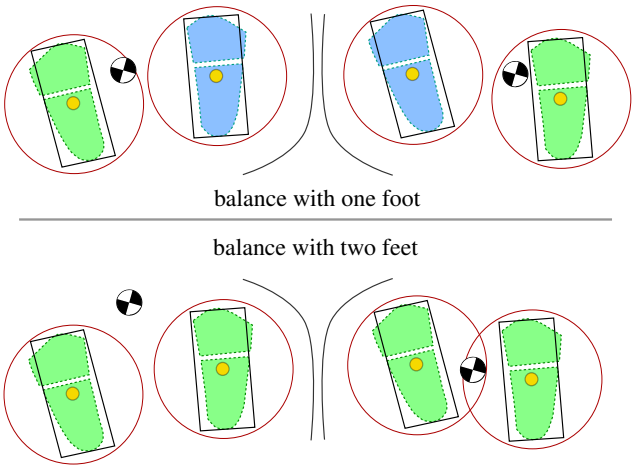

Figure 4: *Esquema da determinação do pé de apoio quando o personagem não segue um movimento capturado. Os raios das circunferências em vermelho (zona de influência) são determinados pelo usuário, cujo centro é o meio do pé do personagem. Se a projeção do COM ($com\perp$) estiver dentro desta zona de influência do pé considera-se o contato deste pé com o solo (ficando verde) ou se $com\perp$ não estiver em nenhuma das duas zonas de influência. Caso a $com\perp$ não estiver em uma zona de influência e estiver na outra zona de influência o pé fica no ar (ficando azul).*

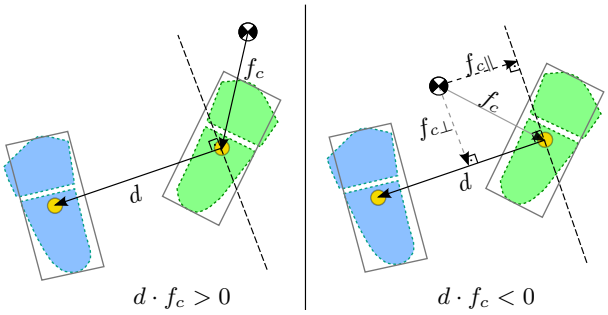

Figure 5: *Quando a força de controle (f_c) e o vetor de direção do pé que encontra-se em apoio ao pé que está no ar (d), ambos projetados no solo, possuem a mesma direção, f_c não será alterado. Caso contrário f_c será projetado no vetor d, utilizando assim a força de controle paralela ($f_{c\|}$) ao vetor d, fazendo com que o pé que esteja no solo possa se tornar pé de apoio de forma mais natural.*

onde, d_u é o vetor unitário determinado pela posição do pé de apoio ao pé que está no ar projetado no solo. Desta forma, a transição de um pé para dois pés fez com que o movimento parecesse mais natural. Sendo assim, o personagem pode trocar as hierarquias de Jacobiana mesmo sem seguir um movimento capturado.

Ao trabalhar com movimentos de referência, observamos em alguns casos que nos movimentos com locomoção o personagem chega a "arrastar" o pé no chão mesmo quando ainda executa um movimento, desta maneira não existe uma exatidão para determinar quando o pé deve considerar-se no "ar" ou apoiado no solo. Esses casos poderiam levar uma descontinuidade do movimento do pé no personagem simulado. Observando isso, o sensor utilizado para determinar qual o pé que está apoiado, ao seguir um movimento

de referência, leva-se em consideração dois quesitos: quando o pé pode se desconectar do solo e quando o pé pode se conectar ao solo, ou seja, ser configurado como pé de apoio. No primeiro quesito é necessário somente as informações adquiridas do movimento capturado, no segundo são utilizadas duas informações em conjunto, informações adquiridas do movimento de referência e das GRFs de contato do pé do personagem simulado.

Os movimentos de referência não trazem consigo dados que determinam qual o pé que está em contato com o solo, para complementar esta carência determinou-se manualmente uma estrutura que dita qual o pé que possa estar em contato, ou não, com o solo em cada *frame* do movimento capturado. Essa informação se faz necessária para se adquirir simulações mais comportadas de acordo com o movimento a ser seguido, vendo-se que a técnica proposta necessita, para que seu controle funcione da melhor forma possível, da informação de que pé dever ser considerado de apoio. No caso do pé desconectar-se do solo utilizamos as informações dessa estrutura, caso o pé não seja considerado pé de apoio na estrutura, a fricção deste pé, no personagem simulado, é zerada fazendo com que o pé deslize no solo, em algumas situações. Para determinar qual pé deve estar contato com o solo é verificado primeiramente se o pé na estrutura está em contato com o solo, naquele determinado instante, caso esteja observa-se a soma das GRFs no eixo y, se for maior que zero o pé do personagem é considerado pé de apoio. Caso um dos requisitos não seja verdadeiro o pé é não é considerado como pé de apoio na simulação.

Com base no controle de equilíbrio, simplificação de contato e no esquema de sensores elaborou-se uma série de testes e os resultados podem ser conferidos na seção seguinte.

7 Results

The experiments have used a humanoid character model with mass of approximately 72kg. Open Dynamics Engine (ODE) [Smith 2014] has been used for the simulation, with gravitational acceleration g=-9,8m/s^2, friction coefficient μ=1.0, and simulation step 0.0005s [Coros et al. 2010]. Ground contact have been simulated with *Error Reduction Parameter* (ERP) and *Constraint Force Mixing* (CFM) equals to ERF = 0.02 and CFM = 0.0001. A mesh has been used only for rendering (Figure 6b), and does not influence over the physical simulation at all. All simulations have been performed in real time on a 2.20GHz x 8 Intel Core i7 machine.

In this work, the proposed balance strategy allows the character to follow captured motions or static poses, obeying some goals imposed by the user. A series of experiments was performed to demonstrate some of the features and applications of the control structure. An accompanying video is available for better visualization and evaluation of the results. In all tests, the simulated character has 13 spherical joints, totaling 39 internal DOFs, and boxes are used as the geometries of all links (including the feet) (Figure 6a). The video shows the character in different situations, in both double and single stance, with two feet on the ground and with a single support foot.

Virtual control terms. The influence of each term of the virtual controller over balance has been tested: control force, gravity compensation, linear momentum error, control torque and angular momentum error. These terms directly influence the character's balance and were based on other works. As some examples, [Macchietto et al. 2009] used angular momentum, [Geijtenbeek et al. 2012] [Zordan and Hodgins 2002] used the goal of keeping the COM over the center of a support region, and [Coros et al. 2010] used individual Jacobians to compensate gravity, though not in a unified manner, as in this work.

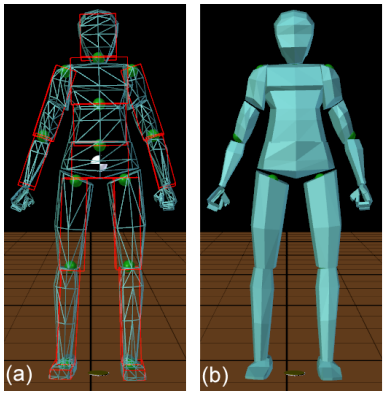

Figure 6: *Character used in the tests. (a) The red wireframe corresponds to the shape of the character's links, as considered in the simulation. The green spheres represent the joints, which are all spherical. (b) Character's mesh used only for rendering.*

Tracking captured motions. To track reference motions, PD controllers were used in the joints. However, the goal of the PD controllers may conflict with the goals of the balance control. In order to show that these two components of the controller adapt well while tracking the captured motions, Figure 7 shows the graphs of the angles (axis x, y and z) of the character's right hip as an example, comparing the captured and simulated motions. Note that, as in [Geijtenbeek et al. 2012], we can use the captured motion directly, without any adaptation to the control proposed in this paper.

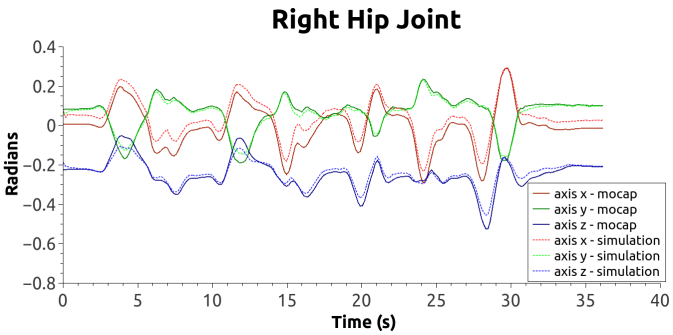

Figure 7: *Angular difference, at the right hip joint, between captured and simulated motions. In the considered motion, the character is throwing punches (see the reference video).*

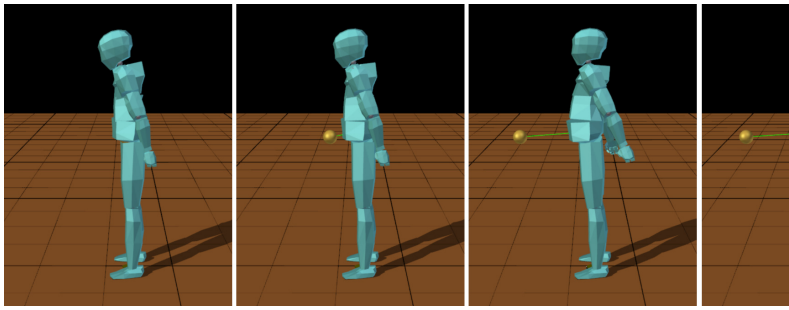

Figure 8: *Response of the balance control in the presence of an external stimulus. Using all links of its body, as a whole, the character automatically keeps the center of mass (black and white ball) over the support area while its pelvis is moved by an external controller.*

External disturbances. In performed tests, the character was disturbed in two ways: through external forces directly applied and through collisions with objects thrown at him. External forces were applied in standing position, focusing on the overall motion of all links working together in the balance recovery. The objects were thrown on random parts of the character, in order to observe its behavior while tracking captured motions. The thrown objects were spheres with density of $100kg/m^3$ and velocity varying between $7m/s$ and $15m/s$. The density was gradually increased to the point that the character could not remain standing (approximately in the density of $125kg/m^3$).

External controllers. In order to observe further the influence of all links of the character to maintain balance, an external positional controller (Equation 24) was used. This controller operates at each instant of the simulation by applying an external force on a given link of the character so that the COM of this link reaches a virtual point, chosen by the user. The stability provided by the simplified contact allows the animator to easily tune the constants of the PD controllers, of the balance control and of these external controllers, according to his purpose. The Figure 6 shows an example in which the animator activates one of these external controllers in order to move the pelvis of the character to front. Note that the balance control overlaps, clearly causing the character to be worked as a whole to keep the COM over the support region. The external force applied by the external positional control has the following expression:

$$f_{externa} = pks(pv - efetor) - pkd(\omega_{link}), \qquad (24)$$

where pv is the position of the virtual point informed by the animator, $effector$ is the current position of the COM of the link of the character which must reach the pv, ω_{link} is the linear velocity of this link, and pks and pkd are user-chosen constants.

Holding cup. The animator can also restrict the orientations of character's specific links. In this test, an extra internal PD controller was used at the character's wrist in order to restrict, in global coordinates, the link that represents its hand. Note that only the rotations in the x and z axes are controlled so that they do not turn the cup (cup is free to rotate on the y axis). It was observed that the character could both track captured motions and balance itself while still respecting this additional constraint (see the reference video).

Simplified contact. Some tests were performed in order to specifically verify the contribution of using the proposed simplified contact model (see the reference video). The first test shows that the initialization of the simulation is greatly facilitated. Without the proposed model, the animator would need to finely adjust the initial pose of the character or use optimization, as in [Geijtenbeek et al. 2012]. Moreover, soon after it is initialized, the character becomes stable enough to balance itself even without the simplified contact. The second test compares the character's behavior under external intervention of the environment, with and without using the simplified contact, testifying the stability provided by the proposed simplification again. The third test shows that, if preferable, it is still allowed to the animator to sacrifice a bit of physical correctness of the simulation in exchange for even greater stability, by just adjusting the parameters of the simplified contact.

8 Discussion and Future Work

The ease of playing captured motions directly, without worrying about the stability of a character, is clearly a huge draw for animators that produce games, movies and virtual environments in general. However, the treatment of balance is one of the main difficulties when trying to add physics to the environment and, of course, to the characters. The requirement of additional knowledge, often involving complicated issues to deal with, such as when using optimization, greatly discourages the use of physics, especially for novice animators. We believe that the convenience provided by our simplified contact model is the main contribution of this work. The ensured stability and the ease with which the torque compensation on the supporting foot is implemented should shorten the transition from the approaches that produce purely cinematic animation to those that produce physically based animation. Thus, this paper aims at helping animators to migrate from purely cinematic approaches to dynamic approaches.

It is fair to say that, although the proposed approach has a useful practical appeal, trying to simulate accurately the morphology of animals in general (not just humans) is still essential, and obviously deserves a lot of research. In this sense, optimization has proven to be very useful [Nunes et al. 2012] [Geijtenbeek et al. 2012] [Wang et al. 2012] [Geijtenbeek et al. 2013]. For future work, we believe that the proposed model can be used to improve the search space based on the work of [Van De Panne and Lamouret 1995], but more effectively. The idea would be to minimize the artificial compensation torque on the supporting foot, and to get an optimized result that works without the simplified contact. In other words, if the objective function is well behaved because of the stability provided by the simplified contact, the optimizer would be directed to a shorter and safer path (with less local minima) to achieve the final result.

.1 Spatial Vector Notation

The *spatial vector notation* is based on [Cline 1999; Nunes 2012], and is used throughout the formulation related to the Jacobian. Basically, this notation consists in concatenating angular and linear information in vectors with six coordinates. The main advantage is that such grouped information may be transformed between different coordinate frames in an unified way, by using the *adjoint matrices*.

A *spatial velocity*, also known as *twist*, is represented by the vector $\phi = (\omega^T \ v^T)^T$, where ω corresponds to an angular velocity and v corresponds to a linear velocity. Similarly, a *spatial force*, also known as *wrench*, is represented by the vector $w = (\tau^T \ f^T)^T$, where τ corresponds to a torque and f corresponds to a force.

The adjoint matrix, ${}^b_a Ad$, with dimension 6×6, transforms a spatial velocity, ${}^a\phi$, given in coordinates of a, at a corresponding spatial velocity, ${}^b\phi$, given in coordinates of b: ${}^b\phi = {}^b_a Ad \ {}^a\phi$. ${}^b_a Ad$ can be easily constructed from the corresponding *homogeneous transformation matrix* ${}^b_a T$:

$$ {}^b_a T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}, \quad {}^b_a Ad = \begin{bmatrix} R & 0 \\ [p]\,R & R \end{bmatrix}, \tag{25} $$

where $[p]$ corresponds to the skew-symmetric matrix, with dimension 3×3, equivalent to the cross product $p\times$:

$$ [p] = p\times = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}, \tag{26} $$

where p_x, p_y and p_z are the coordinates of the vector p. Note that the inverse transpose of ${}^b_a Ad$ transforms *spatial forces*: ${}^b w = {}^b_a Ad^{-T} \ {}^a w$. Namely, its transpose transforms *spatial forces* of b to a: ${}^a w = {}^b_a Ad^T \ {}^b w$.

The mass of a given system a may also be defined using the spatial vector notation. A matrix with dimension 6×6 is used to group its inertia $\mathcal{I}_a$ and its mass m_a:

$$ {}^b_a M = \begin{bmatrix} {}^b\mathcal{I}_a & 0 \\ 0 & m_a I \end{bmatrix}, \tag{27} $$

where ${}^b_a M$ is the *spatial mass* of a in the coordinates of b, and I is the identity matrix with dimension 3×3.

References

ABE, Y., DA SILVA, M., AND POPOVIĆ, J. 2007. Multiobjective control with frictional contacts. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '07, 249–258.

CLINE, M. B. 1999. *Rigid Body Simulation with Contact and Constraints*. Master's thesis, The University of Texas at Austin.

COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2010. Generalized biped walking control. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 130:1–130:9.

GEIJTENBEEK, T., AND PRONOST, N. 2012. Interactive character animation using simulated physics: A state-of-the-art review. *Comp. Graph. Forum 31*, 8 (Dec.), 2492–2515.

GEIJTENBEEK, T., PRONOST, N., AND VAN DER STAPPEN, A. F. 2012. Simple data-driven control for simulated bipeds. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '12, 211–219.

GEIJTENBEEK, T., VAN DE PANNE, M., AND VAN DER STAPPEN, A. F. 2013. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics 32*, 6.

JAIN, S., AND LIU, C. K. 2011. Controlling physics-based characters using soft contacts. *ACM Trans. Graph. (SIGGRAPH Asia) 30* (Dec.), 163:1–163:10.

LEE, Y., KIM, S., AND LEE, J. 2010. Data-driven biped control. In *ACM SIGGRAPH 2010 Papers*, ACM, New York, NY, USA, SIGGRAPH '10, 129:1–129:8.

MACCHIETTO, A., ZORDAN, V., AND SHELTON, C. R. 2009. Momentum control for balance. In *ACM SIGGRAPH 2009 papers*, ACM, New York, NY, USA, SIGGRAPH '09, 80:1–80:8.

NUNES, R. F., CAVALCANTE-NETO, J. B., VIDAL, C. A., KRY, P. G., AND ZORDAN, V. B. 2012. Using natural vibrations to guide control for locomotion. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, I3D '12, 87–94.

NUNES, R. F. 2012. *Usando Vibrações Naturais na Descrição e no Controle de Locomoções Fisicamente Simuladas de Personagens Articulados Arbitrários*. PhD thesis, Universidade Federal do Ceará - UFC.

POPOV, V. 2010. Coulomb's law of friction. In *Contact Mechanics and Friction*. Springer Berlin Heidelberg, 133–154.

PRATT, J. E., CHEW, C.-M., TORRES, A., DILWORTH, P., AND PRATT, G. A. 2001. Virtual model control: An intuitive approach for bipedal locomotion. *I. J. Robotic Res. 20*, 2, 129–143.

SMITH, R., 2014. http://www.ode.org/.

SUNADA, C., ARGAEZ, D., DUBOWSKY, S., AND MAVROIDIS, C. 1994. A coordinated jacobian transpose control for mobile multi-limbed robotic systems. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1910–1915 vol.3.

VAN DE PANNE, M., AND LAMOURET, A. 1995. Guided optimization for balanced locomotion. In *6th Eurographics Workshop on Animation and Simulation, Computer Animation and Simulation, September, 1995*, Springer, Maastricht, Pays-Bas, D. Terzopoulos and D. Thalmann, Eds., Eurographics, 165–177.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2009. Optimizing walking controllers. In *ACM SIGGRAPH Asia 2009 papers*, ACM, New York, NY, USA, SIGGRAPH Asia '09, 168:1–168:8.

WANG, J. M., HAMNER, S. R., DELP, S. L., AND KOLTUN, V. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Trans. Graph. 31*, 4 (July), 25:1–25:11.

WROTEK, P. 2006. Dynamo: Dynamic data-driven character control with adjustable balance. In *in Proc. Sandbox Symp. Video Games, Boston,MA, USA (ACM*, Press, 61–70.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: simple biped locomotion control. In *ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, SIGGRAPH '07.

ZORDAN, V. B., AND HODGINS, J. K. 2002. Motion capture-driven simulations that hit and react. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, New York, NY, USA, SCA '02, 89–96.