DOI: 10.1111/cgf.13096

COMPUTER GRAPHICS *forum*
Volume 00 (2017), number 0 pp. 1–12

Tunable Robustness: An Artificial Contact Strategy with Virtual Actuator Control for Balance

D. B. da Silva[1], R. F. Nunes[1], C. A. Vidal[1], J. B. Cavalcante-Neto[1], P. G. Kry[2] and V. B. Zordan[3]

[1]Federal University of Ceará, Brazil
{danilobs, rubens, cvidal, joaquim}@lia.ufc.br
[2]McGill University, Canada
kry@cs.mcgill.ca
[3]Clemson University, USA
vbz@g.clemson.edu

Abstract

Physically based characters have not yet received wide adoption in the entertainment industry because control remains both difficult and unreliable. Even with the incorporation of motion capture for reference, which adds believability, characters fail to be convincing in their appearance when the control is not robust. To address these issues, we propose a simple Jacobian transpose torque controller that employs virtual actuators to create a fast and reasonable tracking system for motion capture. We combine this controller with a novel approach we call the topple-free foot strategy which conservatively applies artificial torques to the standing foot to produce a character that is capable of performing with arbitrary robustness. The system is both easy to implement and straightforward for the animator to adjust to the desired robustness, by considering the trade-off between physical realism and stability. We showcase the benefit of our system with a wide variety of example simulations, including energetic motions with multiple support contact changes, such as capoeira, as well as an extension that highlights the approach coupled with a Simbicon controlled walker. With this work, we aim to advance the state-of-the-art in the practical design for physically based characters that can employ unaltered reference motion (e.g. motion capture data) and directly adapt it to a simulated environment without the need for optimization or inverse dynamics.

Keywords: 3D interaction, physics-based animation, motion control

ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Control research for simulation often presents important trade-offs between controller robustness and simplicity, as well as computational cost and interactivity. Further, in physics-based synthesis for animation, issues of controllability can be at odds with naturalness, especially as simulated character control systems can be instrumented to include 'super-human' components like perfect sensors, zero latency (or even perfect prediction [NVCNZ08]), as well as infinite strength. As simulated characters become more *supernatural*, they can become arbitrarily robust. However, special care must be placed to keep characters from becoming *unnatural*. This paper proposes a new balance among these important trade-offs which aims to build simple, fast-to-compute controls that are made to be tunably robust through the use of limited external control inputs.

Specifically, we introduce a robust controller for balance behaviours that combines a custom Jacobian transpose control that is fast to compute with a novel 'topple-free' foot (TFF) model that gives the character arbitrarily robust balance-ability at the (potential) cost of naturalness. We design our foot model to be easily tunable so that an animator can intuitively gain access to a spectrum of solutions from perfectly realistic, to plausibly realistic, to super-human standing balance. The Jacobian transpose controller combines powerful aspects of many recent controllers, including full-body position and rotation control, but circumvents the need for heavy computation found in most related efforts. The combination allows us to generate

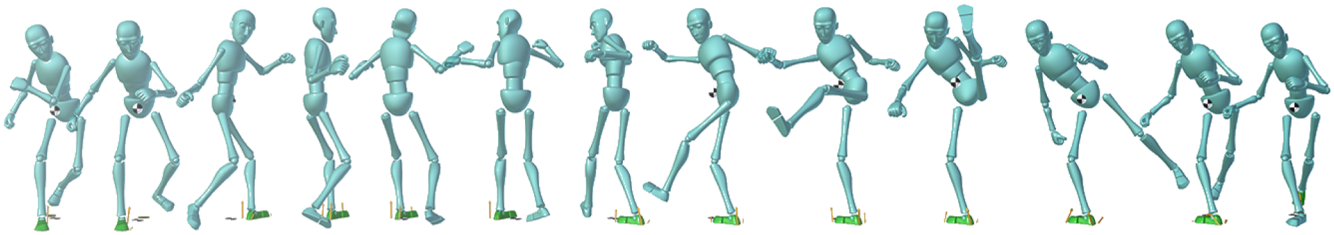

Figure 1: *Our method allows a physically simulated character to directly track unaltered reference motions, with no need of any type of optimization, even for energetic motions with support contact changes.*

balanced motion, even in extremely challenging examples such as the capoeira motion shown in Figure 1.

While the related literature provides many examples of how simulated characters can be limited to better match human capabilities, such as adding actuator noise [WFH10], torque limits, control latencies [ZMCF05], and human-like actuators [WHDK12, GvdPvdS13], the use of supernatural (artificial) forces to improve control remains an attractive option to many as well [AvdP13]. Within this spectrum, our choice is to make available a deliberately limited set of artificial control torques applied to the feet in order to better 'support' standing balance. This choice is purposeful in that, in standing balance activities, the means of control are derived expressly through the ground reaction forces (GRF) passed through the foot (or feet), and by carefully augmenting the foot's ability in a limited manner, we aim to strike a desirable compromise between adding robustness while not making the character appear supernatural.

Further, how foot geometry, its articulation, and contact are modelled grossly effect balance. However, the foot models used for simulations are most often overly simplified and, therefore, cannot reproduce the same distribution of GRFs for real feet, which are structurally complex and flexible due to soft tissues. Jain and Liu [JL11] have shown that increasing the complexity of the foot model makes an existing full-body controller more robust because it is able to better access a broader distribution of GRFs. In this paper, we propose the use of an artificial contact strategy to facilitate robustness for a simple balance controller, without the need of a sophisticated foot model. We believe in the importance of modelling a realistic foot to yield the same range of redistribution of contact forces of a real foot [JL11]. However, such a model still requires a great computational effort. Therefore, we propose a much simpler solution, which consists of uncoupling the foot's geometry from the foot's physical interaction with the ground (see Figure 2).

As such, the contributions of this work include: (1) a unified Jacobian transpose control approach that combines position and rotation through a direct, fast computation; and this is coupled with (2) a tunable, *super*-robust contact model that automatically arbitrates between supplying only the torques necessary to remain balanced and supplying supernatural torques that degrade the naturalness of the resulting motion.

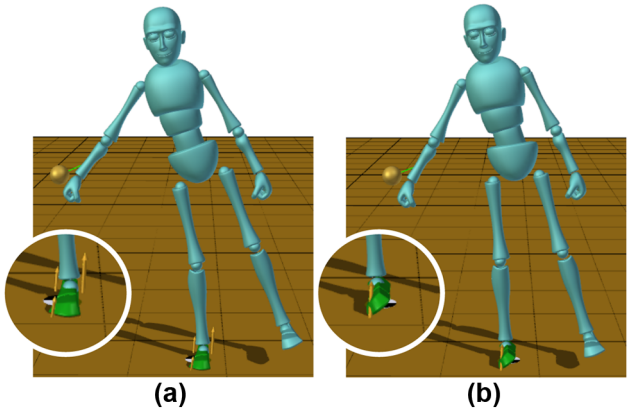

Figure 2: *(a) The proposed simple contact strategy increases the stability between foot and ground, even when the foot's representation (e.g. geometry, softness) is inadequate. (b) Without the proposed strategy, the internal torque applied to the foot is not correctly compensated. In this case, more complex controllers involving optimization [MZS09, GPvdS12] would be required.*

2. Related Work

Our work is inspired by research in multi-objective (MO) control [AdSP07, MZS09, WFH09, GPvdS12, MWTK13] (among others.) One goal of MO control is to take advantage of the activator redundancy in a multi-linked structure in order to guide a select set of high-level objectives (e.g. centre of mass, angular momentum). We share this effort in our controller. However, a major disadvantage of the MO approach is that it relies on computationally costly optimizations. Less frequent optimization and planning have been employed to offset this (e.g. in work by da Silva *et al.* [dSAP08]), but the computational hit without guaranteed robustness has made the work infeasible for commercial applications to date. Our strategy sidesteps this issue through the use of a directly computed *fast* Jacobian-transpose controller with much of the same structure of these various works, but without the need for an expensive solver. Further, our unique foot strategy allows the resulting behaviour to have arbitrary robustness. Together, the combination proposed in this paper sacrifices a degree of physical realism in a tunable manner, in exchange for fast and robust control.

Recent works synthesize full-body motions by using methods that exploit reference motion. Very interesting results can be produced when data are used because it provides both subtle details and a reliable foundation for building new motions. Han *et al.* [HENSfSYS16] address the challenge of reproducing reference motions using off-line trajectory optimization with non-physical forces.

Liu *et al.* [LvdPY16] present a method for learning robust feedback strategies that does not need non-physical forces. With a costly offline optimization they build a control graph from reference clips and transitions between clips, and the result is an excellent real-time character controller with all the diverse skills included in the reference motions. In contrast, our method has a straightforward implementation that provides an online solution for tracking reference motions without a costly optimization.

We are not the first to employ a Jacobian transpose strategy to control characters. For locomotion control, Coros *et al.* [CBvdP10] incorporate a Virtual Actuator (VA) form of Jacobian transpose following other recent work [SADM94, PCT*01] along with a balance-strategy [YLvdP07] for motion control to create versatile bipedal locomotion. VA systems have also been used as a basis to provide balance to biped characters [ZH02, GPvdS12]. The strength of VA approaches is that they can be computed and applied without the optimization required for MO control, but have the weakness that the desired values of the VA control are not guaranteed. Previous work avoids this limitation through the careful tuning of control parameters, for example Geijtenbeek *et al.* [GPvdS12] use covariance matrix adaptation (CMA) in an offline optimization to determine gains values for different character body-types. We sidestep the need for careful tuning by relying on our foot strategy. Closest to our VA technique is that of Geijtenbeek *et al.* because it applies a term for both a virtual force and torque about the centre of mass, while others overlook the latter (rotational) control component. However, the formulation of Geijtenbeek *et al.* treats these terms as separate controls, causing potentially competing effects in the resulting behaviour. In contrast, our formulation computes a unified signal from both, resolving control conflict between the two.

Levine and Popović [LP12] offer an approach with a similar goal to our own, prioritizing robustness over physical correctness. However, in their work, they still perform a QP solve which is both complex to implement and expensive to compute. They also strike a different compromise that prevents the character from falling – leading to 'cartoony' motions when large forces are applied. We opt to purposefully allow the character to fall, while maintaining robustness within a tunable bound.

Specifically, our foot strategy adds external torques (not generated from the ground) to create *super*-robust contacts. Ours is not the first to include artificial actuation to improve control. Van de Panne and Lamouret [VL95] add 'Hand of God' forces that aid in balance, while optimizations seek to reduce these values. Wrotek *et al.* [WJM06] employ artificial torques to create reactions to contact forces in motion capture driven characters. While the addition of these artificial torques represents a divergence from realistic simulation, the values we apply are limited to foot torques, and these are carefully controlled to remain small and are also easily tunable.

Several researchers apply special attributes to the foot as a mechanism for improving control. For example, Wang *et al.* [WFH09] increase the number of joints to model a more flexible foot from rigid links, while Jain and Liu [JL11] model the foot as a deformable object in order to improve robustness. Others (including roboticists) use exaggerated foot proportions in order to increase the contact area with the ground and improve balance. Because our strategy separates the foot geometry from the auxiliary control added, our foot strategy provides an alternative with stability that is easily adjustable.

3. Motion Control

Our framework employs a motion controller that negotiates the balance between tracking a desired motion or pose (Section 3.1) and controlling the centre of mass and body orientation via virtual actuators (Section 3.2). The pose controller can either track a reference motion or remain in a desired pose. The virtual actuators act to balance the character by controlling the centre of mass position and whole-body orientation. Further, the full-body control is supported by a novel TFF artificial contact strategy (Section 4) as well as a supervisory control (Section 5) that manages support contacts between the feet in single or double stance. The TFF contact strategy is responsible for artificially adding a compensatory torque to the support foot to provide artificial robustness to the character's stability, when desired.

3.1. Pose control

We use proportional derivative (PD) controllers for tracking reference trajectories of each individual joint. Using quaternions to represent the 3D orientations and considering spherical joints, the PD control torque at a joint is given by

$$\tau_{\text{pose}} = k_p \mathsf{L}(q_r q^{-1}) + k_d(\omega_r - \omega), \tag{1}$$

where q and q_r are the current and the desired (reference) quaternions of the joint, ω and ω_r are the current and desired angular velocities of the joint, and k_p and k_d are user-defined constants. The expression $\mathsf{L}(q_r q^{-1})$ provides the axis scaled by angle representation of the relative orientation of the two quaternions (i.e. the imaginary vector part of the quaternion logarithm). We directly use the motion capture data as the desired orientation and velocity in our PD control. Because each joint of the character connects a child body to a parent body, the joint torques from Equation (1) are applied to the two bodies as equal and opposite.

3.2. Virtual actuator control

Along with the pose controller, we employ *virtual actuators* (VAs) to control centre of mass (CoM) and whole-body orientation. The VA for CoM controls the character's balance directly, while the VA for orientation provides control over whole-body spin, via angular momentum control. Further, the VAs offer convenient methods for gravity compensation and for control of the character's facing direction. To apply the VAs, a Jacobian transpose control is employed.

A Jacobian matrix is typically seen as describing the linear relationship between the velocity of a specific point (e.g. an end effector) and the velocities of the joints that influence the point, according to a predefined hierarchy. For a point on a hand, for example, the Jacobian only involves a sub-chain of articulated links, starting from a fixed base, or root (e.g. the stance foot). For balance, CoM of the character is a useful choice as the end effector, and its velocity depends on the velocities of all bodies (and likewise all joints).

Using the Jacobian transpose to directly compute control torques has the advantage of being very straightforward in comparison to other approaches that solve a quadratic program to account for available contact forces. The disadvantage is that an optimization-based approach can find feasible balance control torques when the Jacobian transpose approach will fail. This motivates the TFF contact strategy in Section 4.

By using *spatial vector notation* (see Appendix A), we determine the Jacobian that relates all joint velocities to both the linear velocity of the centre of mass and the angular velocity of the character as a whole about the centre of mass. We write this as

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

where

$$^{J}\Phi_{J} = \left({}^{j_0}\phi_{j_0}^{T}\,{}^{j_1}\phi_{j_1}^{T}\ldots{}^{j_{n-1}}\phi_{j_{n-1}}^{T}\right)^{T} \tag{3}$$

is the $6n$ dimensional column vector collecting the velocities of each joint, and ϕ_{com} is the angular and linear velocity of the centre of mass. The construction of the centre of mass Jacobian J_{com} is described in detail in Appendix B.

The internal *spatial forces* to apply at the joints of the character are obtained by multiplying the virtual force and torque (see Sections 3.2.1 and 3.2.2) with the Jacobian transpose,

$$^{J}W_{J} = J_{com}^{T}\begin{pmatrix}\tau_{\text{virtual}}\\ f_{\text{virtual}}\end{pmatrix}. \tag{4}$$

Here, the spatial forces are packed in blocks within a $6n$ dimensional vector W, where each spatial force is in the joint's local coordinates, similar to the spatial velocities Φ in Equation (3). That is,

$$^{J}W_{J} = \left({}^{j_0}w_{j_0}^{T}\,{}^{j_1}w_{j_1}^{T}\ldots{}^{j_{n-1}}w_{j_{n-1}}^{T}\right)^{T}. \tag{5}$$

Because only the joint torques are required from each joint's spatial force, we discard the coordinates corresponding to the linear forces.

Finally, the torque to be applied to the joint j of the character, at each instant of the simulation, is computed as

$$^{j}\tau_{total} = {}^{j}\tau_{\text{pose}} + {}^{j}\tau_{VA}, \tag{6}$$

where the pose term comes from Equation (1) and the VA term comes from Equation (4), using only the angular torque of the corresponding joint.

3.2.1. *Centre of mass control*

We control the centre of mass by employing a virtual actuator force with two terms:

$$f_{\text{virtual}} = f_{\text{control}} + f_{g}. \tag{7}$$

The first term is responsible for maintaining the CoM of the character over its support region. We take the common approach of forcing the horizontal CoM position toward a single point p_{sup}, which we call the *support point*. Our choice of the support point (mean of stance feet's contact points) depends on the stance state as detected by the support supervisor (see Section 5). A PD controller *virtually* corrects the horizontal position and velocity of the CoM, as

$$f_{\text{control}} = k_{fp}(p_{rrel} - p_{rel}) + k_{fd}(\dot{p}_{rcom} - \dot{p}_{com}), \tag{8}$$

where $p_{rel} = p_{com} - p_{sup}$ is the relative horizontal position of the CoM of the simulation with respect to the support point, p_{rrel} is similarly the reference (desired) relative horizontal position coming from motion capture (if used), $\dot{p}_{rcom}$ and $\dot{p}_{com}$ are the reference and current linear horizontal velocities of the CoM, and k_{fs} and k_{fd} are user-defined PD gains for the virtual force CoM control. By using relative positions, the controller allows for horizontal translations of the simulated character. If no reference motion is used, then we use $p_{rrel} = 0$ and $\dot{p}_{rcom} = 0$.

The second term is responsible for compensating gravity. We provide a constant vertical virtual force to oppose acceleration due to gravity for the total mass of the character and 'apply' the force at the CoM of the character.

3.2.2. *Orientation control*

To control the angular momentum and whole-body orientation, we introduce a second VA that computes a virtual torque with two terms:

$$\tau_{\text{virtual}} = \tau_{\text{control}} + \tau_{\text{orient}}. \tag{9}$$

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

$$\tau_{\text{control}} = k_{L}(L_{r} - L), \tag{10}$$

where L and L_r are the current and desired (reference) angular momenta, and k_L is a user-defined constant. Note that the momentum is computed as a sum including all links l of the character,

$$L = \sum_{l}\mathcal{I}_{l}\omega_{l} + m_{l}(p_{l} - p_{com}) \times (\dot{p}_{l} - \dot{p}_{com}), \tag{11}$$

where $\mathcal{I}_l$ and ω_l are the inertia and angular velocity of link l, respectively, m_l and p_l are the mass and the CoM position of the link, and p_{com} is the CoM position of the character. When using motion capture, the desired linear and angular velocities of the links define the desired momentum, and we estimate these velocities using finite differences. If motion capture is not used as a reference trajectory, then we simply set the desired angular momentum to zero.

Just as the gravity term in Equation (7) helps maintain the position of the centre of mass, we use a torque to help maintain the overall

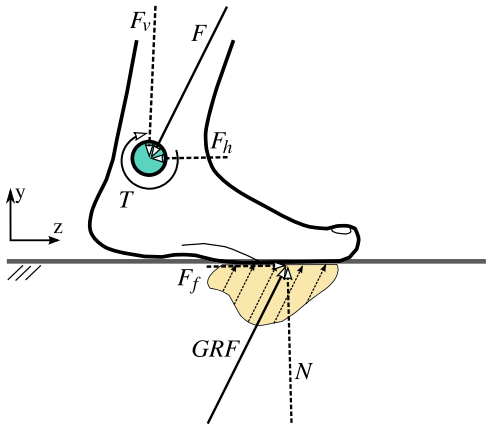

Figure 3: *Forces and torques acting on the foot at rest. The torque, T, at the ankle is balanced by the wrench caused by the GRF which in turn is balanced by the force, F, propagated back through the ankle. If the torque required is large for the available GRF (for example, due to limits in force due to friction, F_f , when the ground is slick) the character's control exerted through the ankle joint will lead to the foot sliding and/or rotating.*

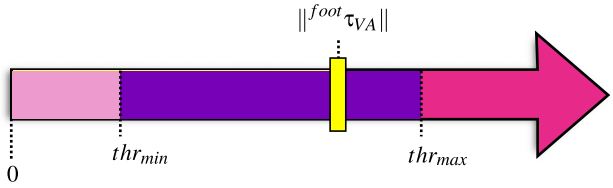

Figure 4: *The thresholds are empirically chosen by the user. Note, as thr_{min} grows, no artificial torque is used. However, if thr_{min} is zero and thr_{max} is very large, the foot is always artificially prevented from toppling.*

orientation (i.e. the facing direction) of the character. To this end, we use the orientation of the chest, and compute the torque as

$$\tau_{\text{orient}} = k_{\tau p}\mathsf{L}(q_r q^{-1}) + k_{\tau d}(\omega_r - \omega), \tag{12}$$

where q and q_r are the current and the desired quaternions of the chest in global coordinates, ω and ω_r are the current and desired angular velocities of the chest, and $k_{\tau p}$ and $k_{\tau d}$ are user-defined PD controller constants. The desired information can be obtained from reference motions or supplied by the user. Instead of applying external torques directly, τ_{orient} is added to the virtual torque which will be converted by the transposed Jacobian into internal joint torques. While different links could be used in this controller, we use the chest for all of our tests. When not using motion capture, we opt to disable this term allowing the character to freely adapt its facing direction.

4. Artificial Contact Strategy

While there are no constraints in the simulation that keep the stance foot fixed, the Jacobian's construction assumes the chain's root (i.e. stance foot) is fixed. Thus, the balancing torques that come from our VA controllers are only effective when GRFs themselves are able to compensate the resultant internal torque applied to the stance foot (see Figure 3). However, this is not always true. When GRFs of the character foot are not able to compensate for the torque, the stance foot may lose contact with the ground or suffer some rotation, and torque control becomes unstable.

At first glance, our solution could be thought of as having an effect similar to adaptively making the feet bigger and heavier in order to provide and maintain stability. But our solution is all about adjusting the torque without changing the force. Changing the mass of the feet and making them larger might be successful for stabilizing a character, but at the cost of motion artefacts that come from managing a larger mass and larger GRFs. Similarly, changing the size of the foot without changing the mass would involve managing the larger geometry and would visibly change how the character interacts with the ground. Nevertheless, our solution allows for larger torques, which can be thought of as being similar to having the torques of a larger foot, but without actually changing the size or mass of the feet.

At high level, we propose the TFF contact strategy as a simple and practical solution that artificially compensates for the excessive torques the controller may require at the stance foot. Intuitively, we know that GRFs can only support torques up to some limit before tipping occurs. Thus, the key idea behind the TFF strategy is to identify such a limit and artificially provide any of the necessary torque beyond this limit. To compute this compensation exactly, we would need to calculate the maximum torque that the GRFs can support at the stance foot, τ_{max}. For example, we might estimate the character at a critical rest state where the GRF is just at the foot's tip and any increasing makes the stance foot rotate. However, for simplicity, we opt not to calculate τ_{max} exactly, and instead, to roughly estimate its value, thr_{min} (Figure 4). Specifically, we let the user choose the desired amount of torque to compensate.

In Figure 4, we propose a straightforward and tunable approach that provides the user with two threshold parameters that tell the system when to help by adding artificial torques to the foot and when to quit, allowing the system a mechanism to give up and fall over. The figure shows three possible zones: while the yellow bar is in the first zone, there is no artificial influence; in the second zone, the artificial influence is enabled; and in the third zone, the character is allowed to fall. The parameter, thr_{min}, in particular, is the lower of the two values chosen by the user and any portion of torque beyond thr_{min} is artificially added to keep the foot from toppling over. Ideally, in order to minimize the use of artificial torques, a large value for thr_{min} is desired. However, since the value is only a rough estimate, by keeping thr_{min} low, a tunable safety margin can be made to assure the foot does not topple. Note that below this value, the controller includes no artificial compensation (for examples, see the accompanying video). To prevent the character from having infinite 'superpowers', the user also empirically chooses a maximum threshold thr_{max} that determines the maximum magnitude possible for the artificial torque.

In our implementation, we artificially add an extra compensatory torque, τ_{comp}, to the stance foot. Specifically focusing on the stance ankle, the internal active torque from Equation (6) applies equal and opposite internal torques on the adjacent bodies, in this case the stance leg and stance foot. Adding the artificial compensatory torque *only to the foot*, we get

$$^{foot}\tau_{total} = {}^{foot}\tau_{pd} + {}^{foot}\tau_{VA} + \tau_{comp}, \tag{13}$$

where τ_{comp} is

$$\tau_{comp} = \begin{cases} 0, & \text{if } \|{}^{f}\tau_d\| \leq thr_{min} \\ -\left(\|{}^{f}\tau_d\| - thr_{min}\right)\frac{{}^{f}\tau_d}{\|{}^{f}\tau_d\|}, & \text{if } thr_{min} < \|{}^{f}\tau_d\| < thr_{max}, \\ \text{falling strategy}, & \text{if } thr_{max} \leq \|{}^{f}\tau_d\| \end{cases} \tag{14}$$

where ${}^{f}\tau_d = {}^{foot}\tau_{VA}$, and $0 \leq thr_{min} \leq thr_{max}$.

Though our contact strategy is able, if desired, to ensure the character's stability even in extreme unbalanced (physically implausible) situations, we also provide a simple falling strategy for the physically plausible cases. When $\| {}^{f\,oot}\tau_{VA}\| \geq thr_{max}$, it is typically desirable to turn off the VA control, allowing the character to fall in a natural manner. When turned off, increasing the damping in pose control also helps to produce a more natural fall.

5. Support Supervisor

Our support supervisor determines stance at each step of the simulation. Specifically, the supervisor sets the character stance state S_C, which encapsulates whether or not each foot is to be considered for support. A generic stance state, S, can be one of the following: $S \in \{left_s, right_s, dual_s\}$. Subsequently, S_C influences the character's hierarchy and its corresponding Jacobians as well as the desired support point and to which foot (or feet) to add artificial torque via the TFF model. Stance feet are treated as the base links of the character's hierarchy influencing all joints outboard from each base (see Figure 7).

We employ reference motion to help guide the choice of the stance state when following motion capture data. Thus, stance state S_{ref} is added to the reference based on the ankle's proximity to the ground, and its linear velocity. To the latter, in our motion capture examples we have many cases (e.g. capoeira and walking) where the foot is in proximity or in contact with the ground, but it is sliding rather than supporting the character. We found that if we also consider the foot's velocity, we can distinguish between incidental contact and stance 'support', as defined for the purpose of control. We identify stance information in the reference once and offline, and also allow manual adjustment as necessary. Further, to aid in simulation of intentional foot sliding or dragging as seen in motion capture, we reduce the simulated friction between sliding foot and ground when this case is identified.

When not using reference motion, we propose a simple automatic strategy to identify contact changes. The basic idea is to define a circular support zone for each foot with a user-defined radius r_z centred at the foot's CoM. This support zone is activated when the physical simulation of the foot is in contact with the ground. If the character's CoM is within either support zone, then the corresponding foot is identified as a 'support' foot, and if both are support feet then S_C is set to *dual* stance. When the CoM is not in either of the support zones, contact alone directly indicates the state for each foot. Figure 5 illustrates the choice of S_C with and without using reference motions.

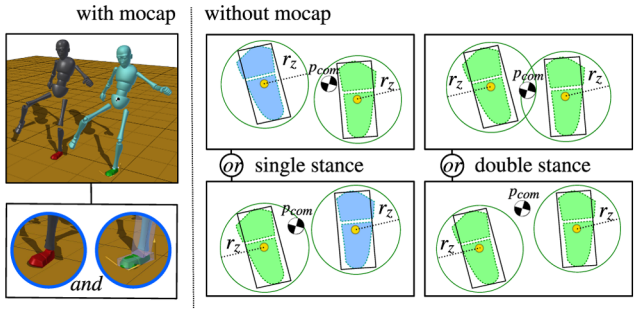

Figure 5: *Operation of the support sensor. Left: With reference motion, S_C considers a specific foot as a stance foot (green) if and only if it is in contact with the ground in simulation and it is a stance foot in S_{ref} (red). Right: Without reference motion, S_C considers a specific foot as a stance foot (green) if and only if it is in contact with the ground in simulation and either p_{com} is in its support zone or p_{com} is outside of both support zones.*

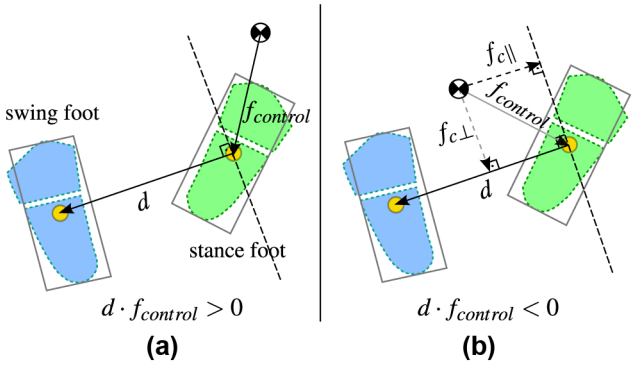

Figure 6: *(a) When the dot product between f_{control} and the vector d from stance foot to swing foot (both ground projected) is positive, f_{control} remains unchanged. (b) Otherwise, f_{control} must be updated to $f_{c\perp}$, allowing the swing foot to step on the ground in a more natural manner. $f_{c\perp} = f_{control} - f_{c\|}$, where $f_{c\|} = (f_{control} \cdot d_u)d_u$ and d_u is the unitary vector of d.*

Based on experimentation, with the support supervisor in place, we opted to make a modification to the virtual CoM control to prevent the controller from naively overlooking the possibility of employing a new support foot. Namely, when the CoM is between the feet and the control is in single stance, we remove the component of f_{control} (Equation 8) parallel to the vector d from the stance foot to the other foot, see Figure 6. This allows the CoM to move toward the non-stance foot (or not) as the dynamics dictate.

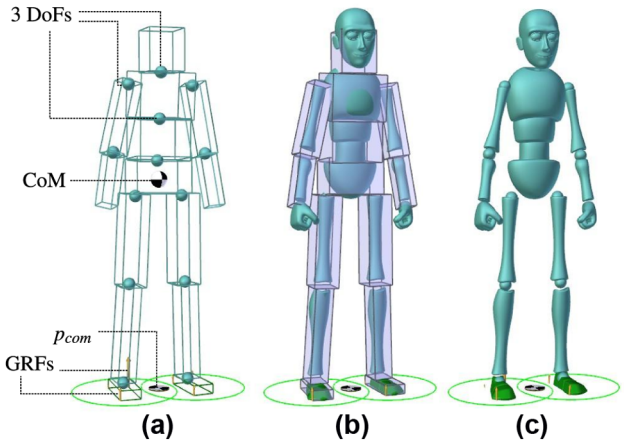

Figure 7: *(a) Wire frame boxes correspond to the shape of the character's links as used in the physics simulation. The spheres represent the joints, with 3 DoFs each. (b) Character's mesh shown with the links. (c) Character's mesh used only for rendering.*

6. Results

In all of our experiments, we use a humanoid character model with mass of approximately 72 kg. The simulated character has 13 spherical joints, totalling 39 internal degrees of freedom (DoFs). Boxes are used as the geometries of all links, including the feet as seen in Figure 7(a), and we use Open Dynamics Engine (ODE) for physics simulation and collision detection. We use a simulation step of 0.0005 s, and by default set the coefficient of friction to $\mu = 1.0$ unless otherwise specified. Ground contacts are simulated with the Error Reduction Parameter (ERP) set to 0.02 and Constraint Force Mixing (CFM) set to 0.0001 based on values used in previous work [CBvdP10]. Figure 7(c) shows the detailed mesh we use for rendering, but this has no influence on the physical simulation. For most of the results, we use a $thr_{min} = 20$ Nm and $thr_{max} = 200$ Nm. All simulations have been performed in real time on a 2.20GHz x 8 Intel Core i7 machine.

We present here a series of experiments that demonstrate the different parts of the control framework, as well as various features and possible applications. Figure 8 shows an overview of the examples discussed in this section. The results show that the proposed balance strategy allows the character to track captured motions or static poses, while obeying goals imposed by the user. Please watch the accompanying video to see examples of the results described below, in addition to examples of some failure cases where we must let the character fall.

Tracking captured motions. To track reference motions, PD controllers often conflict with the goals of the balance control, requiring laborious manual tuning of constants. The stability provided by our contact strategy helps to facilitate the choice of such user-defined constants responsible for making these two components of the controller work well together. We demonstrate the robustness of our framework by directly tracking a variety of unedited reference motion capture sequences, such as punching, kicking, walking, and dancing.

Virtual control terms. The influence of each term of the virtual controller (Sections 3.2.1 and 3.2.2) on the balance can be seen in the supplementary video. This includes the influence of the control force, gravity compensation, momentum error and orientation torque. All of those terms directly influence the character's balance and have been identified as important in other balance control implementations, specifically, the use of angular momentum [MZS09, RvdPK14], the goal of keeping the character's CoM over the centre of a support region [GPvdS12, ZH02], and individual Jacobians to compensate gravity [CBvdP10].

External disturbances. We show examples of the character receiving two types of external disturbances. First, we apply external forces directly to the links, and second, we generate perturbations via collisions with objects thrown at the character. In both cases, the user may choose either a specific link or have a target link selected at random. We vary the direct force magnitudes between 100 N and 300 N, while the thrown objects are spheres with weight varying from 3 kg to 5 kg and have a velocity of 5 m/s.

External controllers. In order to observe further the overall motion of all links working together in the balance recovery, we also use external positional controllers. This type of controller applies an external force on a given link of the character so that the CoM of this link is driven towards a virtual point, which in turn may be interactively driven by the user. The force is computed as

$$f_{\text{external}} = k_{ep}(p_v - p_{\text{effector}}) + k_{ed}(v_v - v_{\text{effector}}), \qquad (15)$$

where p_v and v_v are the position and linear velocity of the virtual point, p_{effector} is the position of the CoM of the link that must reach p_v, the linear velocity of this link is v_{effector}, and k_{ep} and k_{ed} are user-defined PD controller gains. We use $k_{ep} = 550$ and $k_{ed} = 50$ in our experiments, and furthermore, in some tests, we drive p_v according to a desired speed varying between 0.005 m/s and 0.01 m/s. Again, the stability provided by our contact strategy allows the animator to easily tune the constants of the PD controllers, balance control, and external controllers. Figure 9 shows an example in which the animator activates two of these external controllers in order to move the character's arms forward. The balance control dominates the tracking and external PD controllers, and is successful by having the character use all links collectively to keep the CoM over the support region.

Holding cup. An animator can restrict the orientation of specific links in the character. To create a cup holding example we include extra links for hands, and use PD controllers on the wrist to control the orientation in global coordinates, but still with internal torques. Only the rotations in the x and z axes are controlled so that the cup remains horizontal, but is still free to rotate about the vertical axis. As seen in the video, the character can both track captured motions and balance itself while respecting these additional constraints.

Following different directions. When using reference locomotion, we are able to interactively change the forward facing direction to allow for trajectory control. We achieve this by rotating the motion capture data about the vertical axis. The new forward di-

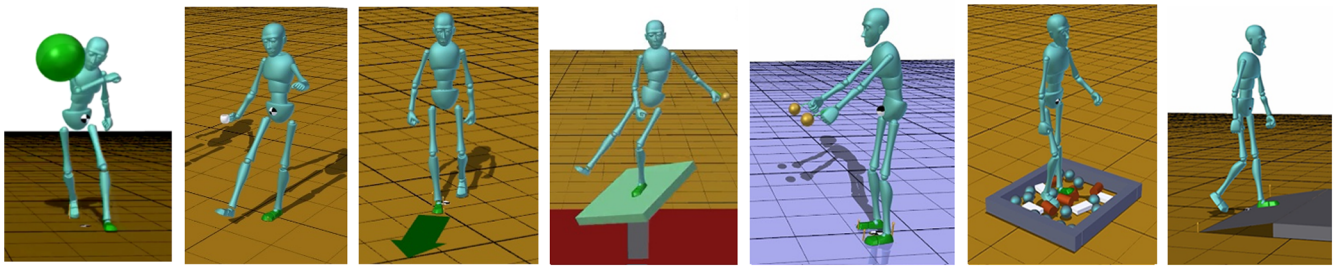

Figure 8: *An overview of our results. In addition to motion capture tracking shown in Figure 1, soft constraints in Figure 9, and SIMBICON inspired virtual force control in Figure 10, from left to right: external disturbances, cup holding with orientation constraints, locomotion following different directions, uneven mobile platform, icy terrain, irregular terrain with loose objects, and ramps (inclines and declines).*

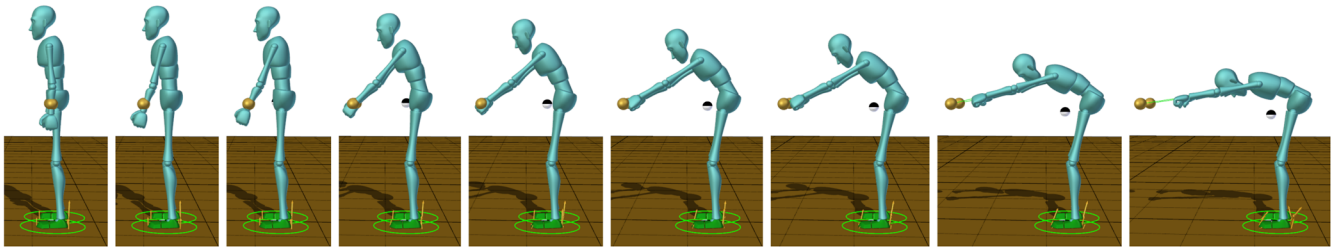

Figure 9: *Response of the balance control in the presence of an external stimulus. The character uses all the links of its body automatically to keep the CoM (shown as a black and white ball) over the support area, while its arms are moved by an external controller.*

rection for the character's coordinate system is set according to a rotation angle around the vertical global axis, from an initial reference forward direction, chosen in world coordinates. Thus, we are able to make the character walk in different directions as seen in the accompanying video.

Simple contact strategy. We performed several tests to verify the contribution of our proposed contact strategy (Section 4). The first test shows that the initialization of the simulation is greatly simplified. Without our contact strategy, the animator would need to make fine adjustments to the initial pose of the character or use optimization, as done by Geijtenbeek *et al.* [GPvdS12]. Moreover, shortly after it is initialized, the character becomes stable enough to balance itself even without our contact strategy. Namely, if we effectively turn off our artificial compensation by setting very high values for thr_{min} (Figure 4), the character is still able to balance under small external disturbances. Finally, we show that the animator may sacrifice some physical correctness of the simulation in exchange for even greater stability, by decreasing the minimum threshold and increasing the maximum threshold for the 'topple-free' foot strategy, making it work for bigger external disturbances.

Changing support base. When not using reference motion, the support sensor described in Section 5 allows the character to automatically identify when a foot is establishing contact with or lifting off the ground. To demonstrate the support sensor we have used external controllers to drive virtual points on the hands that force the displacement of the character's CoM left and right relative to the support zones, thus changing the character stance state S_C.

Platform. We simulate a platform that rotates around three axes in order to produce a very challenging balance scenario. With the character standing on the platform, we observe its behaviour while we change the orientation of the structure. Even with large and fast rotations, the character is able to maintain balance on the platform by adjusting its posture such that the projection of its CoM remains near the desired position.

Surface properties. We observe that our contact strategy adapts well even when the ground's surface properties are changed. In one example, we lower the coefficient of friction of the ground to make it slippery as ice ($\mu = 0.09$) and pull the character by its hands using external controllers. The results show that the character can successfully maintain balance by shifting its weight while sliding. We find that using larger values than $\mu = 0.09$ makes the character bend further while sliding. In another example, we change the slope of the ground. Though an unedited reference motion can be used for small slopes between -10° and 10°, we find that some motion adaptation (e.g. through optimization [WFH09]) is necessary in order to get better results for larger slopes. Simple strategies for controlling the height of the swing foot should also help, and is a topic that could be investigated in future work.

SIMBICON. We use an adaptation of the balance method developed by Yin *et al.* [YLvdP07], called SIMBICON, to demonstrate how such walking controllers can use our simple contact strategy with virtual forces. SIMBICON uses the swing foot to update the character's support base for the next step, according to the character's CoM velocity and the projected distance between the stance foot and the character's CoM. In our adaptation, we use virtual

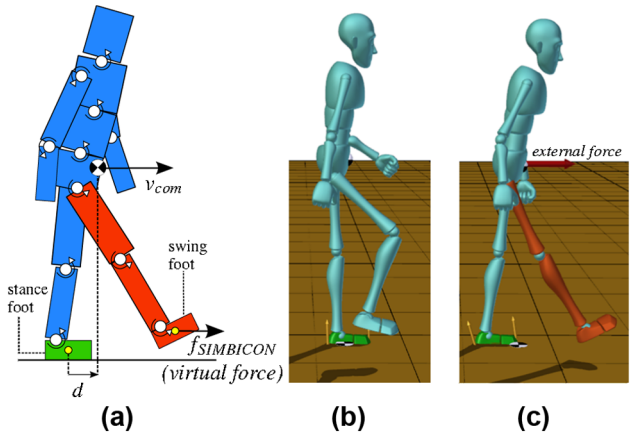

Figure 10: *SIMBICON with virtual force. (a) Virtual force is 'applied' to the swing foot, based on a Jacobian's construction that uses the whole structure of bodies that make up the swing leg (red leg). (b) If no external force is applied, the SIMBICON is not used. (c) When an external force is applied on the character, the SIMBICON acts on the swing leg updating the support base.*

forces as shown in Figure 10 to provide a SIMBICON style balance control for walking. Such virtual forces are 'applied' to the swing leg (Figure 10 a), and are calculated following the SIMBICON inspired feedback model

$$f_{\text{SIMBICON}} = k_{sp}\, d + k_{sd}\, v_{com}, \tag{16}$$

where, k_{sp} and k_{sd} are user-chosen PD controller constants, d is the ground projected distance of the character's CoM ahead of the support foot's CoM, and $v_{com} = \dot{p}_{com}$ is the linear horizontal velocity of the character's CoM. Rather than applying it as an external force, this virtual force is converted into internal torques that we apply on the swing leg, along with the PD control torques, while balance control torques τ_{bal} on the swing leg are suppressed. We build a specific extra Jacobian affecting only the swing leg for this purpose, considering the pelvis as the root and the swing foot as the end effector. With this control structure, the character can move in all directions using only one reference motion (walking in place). We observe that the motion is robust to the application of external forces ranging from 100 to 120 N for 0.2 to 1 s.

Irregular ground. Even with all the graphics realism, one of the most noticeable problems in recent games is the footskate [KSG02]. The difficulties arise when using motion capture directly, for instance on different terrain, or at transitions between different motion clips. We demonstrate a particularly challenging motion capture reuse scenario of walking in place on irregular and loose terrain. We note that our method conveniently allows the character's feet to adapt both in position and orientation in an automatic manner according to the physical environment. To simulate an irregular ground, we place objects with different shapes on the ground, in a limited area (see second from right snapshot in Figure 8). While tracking a walking-in-place motion capture clip, the character is able to maintain balance and naturally interact with the objects below its feet. Finally, similar to the inclines and declines in the surface properties example, we also demonstrate the character's ability to walk over a small ramp.

7. Conclusions and Future Work

This paper aims to help animators migrate from kinematics (e.g. keyframe or motion capture) to physically based approaches for characters in video games, animation, and virtual reality environments. While physics can be attractive because of how it provides spontaneity and variety to movement, it can be unwieldy and require complex solutions for control. Thus, character application developers opt against the use of physical simulation in lieu of more manageable techniques. The main contribution of this paper is to close this gap by offering a simple Jacobian transpose controller with the ability to easily produce reliably robust physically plausible motion through our TFF contact strategy.

While our approach has practical appeal, our long-term goal remains to simulate all aspects of motion accurately. Within this scope, our work reveals a number of limitations and directions for future work. Our near-term research goals include the extension of our method to motions with multiple supports (e.g. quadrupeds, other multi-legged characters, bipeds using arms as support or for holding some prop) or with significant flight phases (e.g. running, jumping, aerial kicks or stunts). A critical piece for such extensions is generalizing and improving the support supervisor as it includes simplifying assumptions related to biped balance behaviours (the main focus of this paper). With regards to the character's morphology, using an extra end-effector (e.g. a hand) as a new support and extending our TFF idea to include the new support seems feasible, although additional tuning will result.

Further, the simplicity of our approach can lead to artefacts that decrease the quality of our resulting motion. While the robust balance does not fail when there are discontinuities in the reference motion, e.g. at the seam in the cyclification of our walk cycle, the pose controller can create visual jittering if the tracked motion is not smooth. Likewise, since the support supervisor is not sophisticated, the control can lead to noticeable artefacts when regular support changes occur, e.g. in walking. Improvement in the support supervisor that smooths its effect would mitigate the latter to an extent.

However, as control and offline parameter optimization remains an attractive direction, one direction for future work includes the use of our proposed TFF strategy to improve search in the spirit of the guided optimization work of van de Panne and Lamouret [VL95]. The idea would be to employ the TFF strategy to stabilize control, and then to gradually minimize such artificial compensation to likely find an optimized result that works without it.

The virtual actuator approach of control provides an alternative to complex and expensive control optimization. This paper helps to highlight the strengths and possibilities of virtual actuators as a control method. We encourage researchers to continue to investigate the approach with or without the stabilization of the TFF contact strategy because we believe it provides a useful foundation for exploring aspects that contribute to naturalness in simulated human movement.

Acknowledgements

We thank the anonymous reviewers for the suggestions for improving the paper. We also thank Sean Burgoon and Arnaldo Vila Nova for providing the character mesh used in the results. This work was supported by CNPq (grant 302726/2016-0 and grant 307941/2013-2), CAPES and NSERC.

Appendix A: Spatial Vector Notation

The *spatial vector notation* is based on that used by Cline [Cli99], 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, to a corresponding spatial velocity, ${}^b\phi$, given in coordinates of b, that is, ${}^b\phi = {}^b_a Ad \, {}^a\phi$. The matrix ${}^b_a Ad$ can be easily constructed from the corresponding *homogeneous transformation matrix* ${}^b_a T$:

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

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

$$ [p] = p \times = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}, \tag{A.2} $$

where p_x, p_y and p_z are the coordinates of the vector p. 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, that is, ${}^a w = {}^b_a Ad^T \, {}^b w$.

The generalized mass of a given body 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, and is diagonal when written in a coordinate frame located at the body's centre of mass and with axes aligned with the body's principal axes of inertia. That is,

$$ {}^b_a M = \begin{bmatrix} {}^b\mathcal{I}_a & 0 \\ 0 & m_a I \end{bmatrix}, \tag{A.3} $$

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

Appendix B: CoM Jacobian Construction

Recall that ${}^J\Phi_J = ({}^{j_0}\phi^T_{j_0} \; {}^{j_1}\phi^T_{j_1} \ldots {}^{j_{n-1}}\phi^T_{j_{n-1}})^T$ is the $6n$ dimensional column vector collecting the velocities of each joint, and ϕ_{com} is the angular and linear velocity of the centre of mass.

Note that there is a close relationship between joints and links. Except for the root link, each link is connected to a parent link by a joint. Thus we can uniquely associate each link to a joint, with the root link being associated with a 6 degree of freedom free-joint that provides the position and orientation of the root in the world. Keeping everything general, we also use a 6-dimensional (angular and linear) velocity of link j with respect to its parent link to express the joint velocity ${}^j\phi_j$ for each internal joint. Note that for rotary and spherical joints, the linear component will be zero because we express the quantity in a coordinate frame fixed to the parent link and located at the joint.

Unlike a simple end effector, the CoM as an end effector is influenced by the position and orientation of all links. Thus, let us show how the velocities of all links, as well as their individual Jacobians, can be proportionally combined, according to their masses, in order to calculate the character's total momentum, relative to its CoM,

$$ {}^{com}\begin{pmatrix} L \\ P \end{pmatrix}_c = {}^{com}\begin{pmatrix} \mathcal{I} \cdot \omega \\ m \cdot v \end{pmatrix}_c = ({}^{com}_c M)({}^{com}\phi_{com}), \tag{B.1} $$

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 subscript denotes quantities that relate to the character as a whole. Matrix ${}^{com}_c M$ can be seen as the total *spatial mass* of the character in the coordinates of its CoM.

As we want to show, the total momentum can also be obtained by the sum involving all links,

$$ {}^{com}\begin{pmatrix} L \\ P \end{pmatrix}_c = \sum_l ({}^{com}_l M)({}^{com}\phi_l) = \sum_l ({}_{com}{}^l Ad)^T ({}^l_l M)({}^l\phi_l), \tag{B.2} $$

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

Each link l has an exclusive Jacobian, J_l, which relates *spatial velocity*, ${}^l\phi_l$, with the vector ${}^J\Phi_J$. 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, that is,

$$ {}^l\phi_l = \sum_j {}^l\phi_j = \sum_j {}^l_j Ad \, {}^j\phi_j = J_l \, {}^J\Phi_J, \tag{B.3} $$

where the sum in j includes all joints that influence the link l. 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:

$$^{l}\phi_l = J_l \, ^{J}\Phi_J = \left[0 \; {}^{l}_{j_1}Ad \; {}^{l}_{j_2}Ad \; 0 \; {}^{l}_{j_4}Ad \; 0 \right] {}^{J}\Phi_J. \quad \text{(B.4)}$$

In order to facilitate the implementation, a given hierarchy may be represented by a table relating all joints (rows) to all links (columns) of the character. Each cell of that table is filled with 0 or 1, according to the hierarchy. Filling a cell with 1 means that the joint corresponding to that row 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 based on the contacts between the feet and the ground, according to our support supervisor (see Section 5).

Finally, combining the Equations (B.1), (B.2) and (B.3), and comparing with the Equation (2), we figure out the final expression of the Jacobian J_{com}, which involves all joints of the character:

$$J_{com} = ({}^{com}_{c}M)^{-1} \sum_{l} ({}_{com}{}^{l}Ad)^{T} ({}^{l}_{l}M) J_l, \quad \text{(B.5)}$$

where

$${}^{com}_{c}M = \sum_{l} ({}_{com}{}^{l}Ad)^{T} ({}^{l}_{l}M) ({}_{com}{}^{l}Ad). \quad \text{(B.6)}$$

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.

References

[AdSP07] Abe Y., da Silva M., Popović J.: Multiobjective control with frictional contacts. In *ACM SIGGRAPH/Eurographics Symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2007), SCA '07, Eurographics Association, pp. 249–258.

[AvdP13] Agrawal S., van de Panne M.: Pareto optimal control for natural and supernatural motions. In *Proceedings of Motion on Games* (2013), ACM, pp. 29–38.

[CBvdP10] Coros S., Beaudoin P., van de Panne M.: Generalized biped walking control. In *ACM SIGGRAPH 2010 papers* (New York, NY, USA, 2010), SIGGRAPH '10, ACM, pp. 130:1–130:9.

[Cli99] Cline M. B.: *Rigid Body Simulation with Contact and Constraints*. Master's thesis, The University of Texas at Austin, 1999.

[dSAP08] da Silva M., Abe Y., Popović J.: Interactive simulation of stylized human locomotion. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 82.

[GPvdS12] Geijtenbeek T., Pronost N., van der Stappen A. F.: Simple data-driven control for simulated bipeds. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2012), SCA '12, Eurographics Association, pp. 211–219.

[GvdPvdS13] Geijtenbeek T., van de Panne M., van der Stappen A. F.: Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics 32*, 6 (2013).

[HENSfSYS16] Han D., Eom H., Noh J., Shin (formerly Sung Yong Shin) J. S.: Data-guided model predictive control based on smoothed contact dynamics. *Computer Graphics Forum 35*, 2 (2016), 533–543.

[JL11] Jain S., Liu C. K.: Controlling physics-based characters using soft contacts. *ACM Transactions on Graphics (SIGGRAPH Asia) 30* (Dec. 2011), 163:1–163:10.

[KSG02] Kovar L., Schreiner J., Gleicher M.: Footskate cleanup for motion capture editing. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2002), pp. 97–104.

[LP12] Levine S., Popović J.: Physically plausible simulation for character animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2012), SCA '12, Eurographics Association, pp. 221–230.

[LvdPY16] Liu L., van de Panne M., Yin K.: Guided learning of control graphs for physics-based characters. *ACM Transactions on Graphics 35*, 3 (2016).

[MWTK13] Mordatch I., Wang J. M., Todorov E., Koltun V.: Animating human lower limbs using contact-invariant optimization. *ACM Transactions on Graphics 32*, 6 (Nov. 2013), 203:1–203:8.

[MZS09] Macchietto A., Zordan V., Shelton C. R.: Momentum control for balance. In *ACM SIGGRAPH 2009 papers* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 80:1–80:8.

[NVCNZ08] Nunes R. F., Vidal C. A., Cavalcante-Neto J. B., Zordan V. B.: Simple feedforward control for responsive motion capture-driven simulations. In *ISVC* (2008), Springer, pp. 488–497.

[PCT*01] Pratt J. E., Chew C.-M., Torres A., Dilworth P., Pratt G. A.: Virtual model control: An intuitive approach for bipedal locomotion. *International Journal of Robotic Research 20*, 2 (2001), 129–143.

[RvdPK14] Rabbani A. H., van de Panne M., Kry P. G.: Anticipatory balance control. In *Proceedings of the Seventh International Conference on Motion in Games* (New York, NY, USA, 2014), MIG '14, ACM, pp. 71–76.

[SADM94] Sunada C., Argaez D., Dubowsky S., Mavroidis C.: A coordinated jacobian transpose control for mobile multi-limbed robotic systems. In *IEEE ICRA* (1994), pp. 1910–1915 vol. 3.

[VL95] Van De Panne M., Lamouret A.: Guided optimization for balanced locomotion. In *6th Eurographics Workshop on Animation and Simulation, 1995* (1995), pp. 165–177.

[WFH09] Wang J. M., Fleet D. J., Hertzmann A.: Optimizing walking controllers. In *ACM SIGGRAPH Asia 2009 papers* (New York, NY, USA, 2009), SIGGRAPH Asia '09, ACM, pp. 168:1–168:8.

[WFH10] Wang J. M., Fleet D. J., Hertzmann A.: Optimizing walking controllers for uncertain inputs and environments. In *ACM SIGGRAPH 2010 papers* (New York, NY, USA, 2010), SIGGRAPH '10, ACM, pp. 73:1–73:8.

[WHDK12] Wang J. M., Hamner S. R., Delp S. L., Koltun V.: Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Transactions on Graphics 31*, 4 (July 2012), 25:1–25:11.

[WJM06] Wrotek P., Jenkins O. C., McGuire M.: Dynamo: Dynamic, data-driven character control with adjustable balance. In *ACM SIGGRAPH Symposium on Videogames* (New York, NY, USA, 2006), Sandbox '06, ACM, pp. 61–70.

[YLvdP07] Yin K., Loken K., van de Panne M.: SIMBICON: simple biped locomotion control. In *ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM.

[ZH02] Zordan V. B., Hodgins J. K.: Motion capture-driven simulations that hit and react. In *ACM SIGGRAPH/Eurographics Symposium on Computer animation* (New York, NY, USA, 2002), SCA '02, ACM, pp. 89–96.

[ZMCF05] Zordan V. B., Majkowska A., Chiu B., Fast M.: Dynamic response for motion capture animation. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, ACM, pp. 697–701.

Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

Video S1

Video S2