Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

# Principal Geodesic Dynamics

M. Tournier[1,2] and L. Reveret[3]

[1]INRIA, CNRS, Université de Montpellier 2
[2]DEMAR-LIRMM
[3]INRIA, CNRS, Université de Grenoble (LJK)

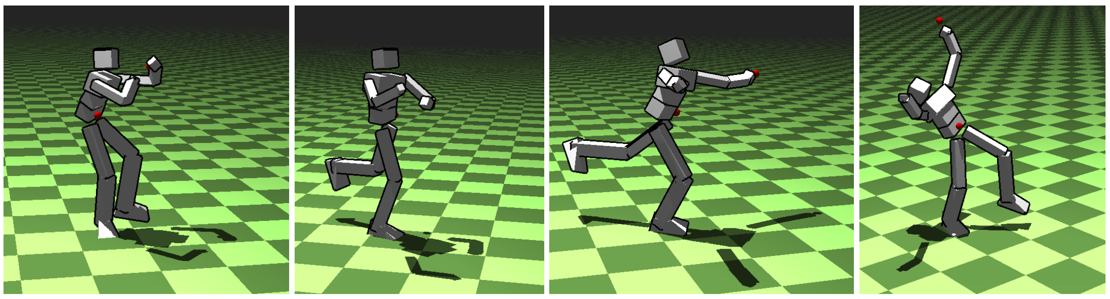

**Figure 1:** *One foot balance. 3 control features are present here: right foot, Center of Mass (CoM), and left hand (red). The system automatically adjusts the left leg position to maintain the CoM above the right foot while the user manipulates the left hand position.*

**Abstract**

*This paper presents a new integration of a data-driven approach using dimension reduction and a physically-based simulation for real-time character animation. We exploit Lie group statistical analysis techniques (Principal Geodesic Analysis, PGA) to approximate the pose manifold of a motion capture sequence by a reduced set of pose geodesics. We integrate this kinematic parametrization into a physically-based animation approach of virtual characters, by using the PGA-reduced parametrization directly as generalized coordinates of a Lagrangian formulation of mechanics. In order to achieve real-time without sacrificing stability, we derive an explicit time integrator by approximating existing variational integrators. Finally, we test our approach in task-space motion control. By formulating both physical simulation and inverse kinematics time stepping schemes as two quadratic programs, we propose a features-based control algorithm that interpolates between the two metrics. This allows for an intuitive trade-off between realistic physical simulation and controllable kinematic manipulation.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: 3D Graphics and Realism—Animation

## 1. Introduction

While human body exhibits numerous degrees of freedom for producing motion, bio-mechanical studies showed they tend to be actuated in a highly-correlated way, resulting in similarly coordinated body motion. This fact has been typically exploited by recent successful approaches based on dimension reduction of human motion. This redundancy offers an opportunity to reduce the total complexity of animating a virtual character, by capturing this structure in a simple parametric model. It allows to generate better-looking animations at a lower computational cost. We propose in this paper an approach to use these reduced parameters directly at the core of a physically-based animation context by using them as generalized coordinates.

In the context of character animation, recent works have shown the potential of mixing dimension reduction with physical simulation. What we propose here is a complete derivation of their integration into a coherent framework working in real-time and with stable control. Our physical simulation includes constraint-based unilateral ground contact model. Constraint-based model is favored over the penalty-based approach to avoid the risk of instability due too strong stiffness. Our approach allows to implement agile change of foot support during simulation, on user request, without planning the contact location explicitly. In order to maintain a real-time simulation while preserving important dynamic invariants, such as angular momentum, we used an explicit integrator based on variational geometric integrators.

To summarize, our contributions are the followings:

- an explicit time integrator for a reduced dimension pose model based on variational geometric integrators for articulated bodies,
- a simple quadratic programming control framework, providing trade-off between physical-simulation and kinematics control using a metric interpolation and real-time animation.

The remainder of the paper is organized as follow:

Section 2 reviews related works in the domain of dimension reduction in the context of both machine learning approach and physically-based animation. Recent works in the domain of task-space control are also discussed. Section 3 presents the algorithm for learning and approximating the pose manifold of a motion capture. Section 4 applies these ideas in the context of physically-based animation of characters using variational integrators. Section 5 on control develops the physical modeling to implement motion control. Examples of controllers for features tracking are presented in 6, leading to results for a balance controller. Finally, benefits and limitations are discussed in section 7.

## 2. Related Work

### 2.1. Dimension Reduction in Machine Learning

Recent advances in the machine learning theory allowed to design systems that capture geometric features as well as modeling time evolution on existing motion capture data. They allow synthesizing plausible motion while still retaining good control.

The idea of using machine learning to compute a higher-level parametrization of a motion given real examples has been widely explored over the past decade. [RCB98] used Radial Basis Functions (RBF) to learn an interpolation space from examples. [MK05] proposed to use geo-statistics to perform better interpolation in a control space. Brand and Hertzmann [BH00] propose a cross-entropy optimization framework to learn HMM for both structure and style from motion capture data. [GMHP04] cast the Inverse Kinematics (IK) problem as the optimization of the likelihood of a Probability Density Function (PDF) over poses, knowing constraints. They propose to learn this PDF using a Gaussian Process Latent Variable Model, a generalization of PCA and RBF models that is computed on motion features (*e.g.* angles, velocities). Then, the corresponding pose likelihood function is optimized under geometric constraints.

All these methods exploit ever more complex non-linear statistical techniques to represent a statistical distribution over poses, possibly using regression against a feature space or directly optimizing the likelihood in which case the computational cost is high.

### 2.2. Dimension Reduction in Physically-based Animation

The benefits of dimension reduction have also been exploited in the context of physically-based animation, since they allow to reduce computational requirements while preserving most dynamic features of the full-dimension simulation. [PW89] pioneered the use of modal analysis in the context of computer graphics. The basic idea of this approach is to decompose a mechanical system with numerous, coupled mechanical degrees of freedom into an equivalent set of mechanically-independent one-dimensional degrees of freedom, usually called *modes*. The set of all modes forms the *modal basis* which can be seen as an alternate representation of the system DOFs.

Due to its numerous advantages, modal analysis has spawned a wide variety of research works ( [JBP06, TLP06, WST09], among others) building on this basic principle. [KRFC09] proposed an extension of this technique to articulated rigid bodies: rather than performing the modal analysis on a mesh-supported, linearized FEM, the authors applied the modal decomposition to the dynamics of an articulated rigid body around a rest pose, obtaining new mechanically-independent angular degrees of freedom around a rest pose. This was the basis for animating locomotion behaviors as combinations of natural vibration modes. Simulation including contacts have been recently proposed in [JL11] and [NCNV*12]. However, in this case, modal analysis has the drawback that the joint stiffness must be known for the modal analysis to make sense. Instead, the reduced basis we obtained is not dependent on hand-chosen stiffness/damping parameters, but only on motion capture data. Besides, expressing the dynamics in our reduced pose parametrization is not restricted to small displacements.

Closer to data-driven statistical approach, [SHP04], and more recently [WMC11], present frameworks where statistical approaches and physically-based animation are integrated. Motion is parameterized with usual kinematics degrees of freedom. In their case, motion manifold learned from PCA acts as a constraint to plausible poses. Such an approach implies an optimization over a period time of several frames. We are proposing a different approach where latent variables induced by manifold learning are directly used as generalized coordinates avoiding a costly optimization.

[YL08] present an alternate physically-based approach featuring dimension reduction for character animation. The goal of this work is to construct a data-driven basis of torques from motion capture data, then extract the coordinates corresponding to the *least* actuated torques. This *near-unactuated* basis is later used to synthesize upper-body, physically-based perturbations on top of existing motion capture animations. The main assumption is that the unactuated coordinates tend to be much more compliant, in the case of external disturbances, than the actuated ones. While this technique shows good results, its drawback compared to expressing the dynamics in the reduced dimension pose space is that it requires inverse dynamics, which is challenging to setup in the case of contacts, probably explaining that the motion perturbation is limited to the upper-body in their context.

## 2.3. Motion Control

The goal of this paper is not to propose new control algorithms, such as ones implementing walking. Our goal is rather to show how control can be implemented using our new formulation of the dynamics of articulated characters. We follow a *task-space control* approach as described in [AdSP07,MZS09,JYL09,LMH10,MLH10]. In this formulation, kinematic goals are formulated in some abstract *task*, or *feature* space, and the system solver is responsible for optimizing these goals using physics laws as a constraint *at each time-step*. This approach can be seen as a synthesis of the advantages of joint-space control and space-time optimization while mitigating their drawbacks: instead of solving a large, global optimization problem preventing interactivity, a smaller, easier problem is solved at each time-step. Besides, the burden of choosing and adapting individual control gains is shifted from the joint space to an abstract task-space.

In [LMH10, MLH10], the kinematic goals are expressed on accelerations, by forming a quadratic objective function. This objective is optimized under the dynamic equations and approximate contact constraints (no complementarity). A *feature* priority solving strategy is proposed, preventing certain control objectives to compete with each others. While this approach is very general and well-grounded for character control, it could arguably be improved especially by the use of a velocity/impulse formulation of dynamics and features, removing the need for second derivative which can be difficult to compute. Last, but not least, a reduced dimension motion model could improve the overall efficiency of this method significantly. These are the points we will address in the remaining of this paper.

## 3. Reduced Manifold for Kinematics

The configuration space of an articulated character, parameterized by its joint orientations, is not an Euclidean vector space. This is the key limitation which prevents from applying safely standard analysis such as PCA. However, the pose manifold $SO(3)^n$ possesses a natural Riemannian structure, for which generalizations of Euclidean algorithms have been proposed, including multi-resolution analysis, interpolation, and statistics. Building on this property, [FLPJ04] proposed an extension of PCA to the case of Riemannian manifolds, first in the case of Lie groups having a compatible Riemannian structure, then on symmetric Riemannian manifolds.

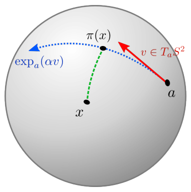

**Figure 2:** *The geodesic projection of point $x$ over the geodesic* $\exp_a(\alpha V)$ (blue) *corresponds to the point where the geodesic segment in green has minimal length.*

Following [TWC*09] in the context of character animation, we have found the PGA algorithm to be a natural, simple dimension reduction technique, suitable for analyzing rotational pose data. We quickly recall main results in the context of character animation in order to introduce notations.

### 3.1. Geodesics on Lie Groups

Geodesics on a smooth manifold arise from a Riemannian *metric*, a smooth operator computing the norm of tangent vectors and the length of curves: geodesics are locally length-minimizing curves. Lie groups, on the other hand, are smooth manifolds where the group operations are smooth. A good introduction to these topics in the context of robot kinematics can be found in [MLS94].

In the case of a Lie group with a compatible Riemannian metric such as $SO(3)$, the metric and algebraic exponential coincide. Given such a Lie group $G$, the geodesic starting at $a \in G$ with initial tangent vector $V \in \mathfrak{g}$ (in body coordinates) is given by $\gamma_{a,V} = a.\exp(\alpha V)$, for $\alpha \in \mathbb{R}$ (*cf.* Figure 2). The geodesic distance between two points $a$ and $b$ of $G$ is then given by:

$$\text{dist}(a,b) = \left|\left|\log(b^{-1}a)\right|\right|$$

In terms of Lie group operations, the projection of $x \in G$ on a geodesic starting at $a \in G$ with body tangent vector $V \in \mathfrak{g}$ is thus:

$$\pi_{a,V}(x) = a\exp(\alpha^\star V) \qquad (1)$$

with

$$\alpha^\star = \underset{\alpha \in \mathbb{R}}{\text{argmin}}\ \text{dist}\left(x, a\exp\left(\alpha V\right)\right)$$

## 3.2. Principal Geodesic Analysis

We now quickly outline the PGA algorithm for orientation data. More in-depth discussions can be found in [FLPJ04, SLHN10] for the general case. In essence, the PGA algorithm generalizes PCA using the geodesic distance and projection instead of the Euclidean ones. Given orientation samples, the first step is to compute their *intrinsic* mean $\mu \in SO(3)$, which minimizes the sum of geodesic distances to the samples [Moa02, Pen06]. The intrinsic mean provides a *coordinate-invariant* description of the data average. In contrast, an Euclidean mean of Euler angles or exponential maps depend on a particular choice of reference coordinate frame. Next, a set of mutually orthogonal geodesic directions is computed by maximizing the geodesic projections of the samples onto the corresponding geodesics.

Dimension reduction is obtained by selecting the $k \in \mathbb{N}$ most significant geodesic directions, where $k$ is usually determined using a variance criterion. As the PGA algorithm only makes use of Lie group operations, its extension to the case of pose data, *i.e.* the product group $SO(3)^n$ with $n \geq 1$, is straightforward.

An approximate computation of the geodesic directions can be obtained by applying a PCA over the log-linearized samples at the intrinsic mean $\mu$. While this approach somehow mitigates the advantages of PGA, we found it more practical to use and sufficient for animation purposes, while retaining the coordinate-invariance of the analysis. The differences between linearized and exact PGA are discussed in detail in [SLHN10].

We now describe how to apply PGA on motion capture data in order to construct reduced dimension skeleton kinematics.

## 3.3. Reduced Kinematics

We begin with an existing motion capture sequence represented as a set of $m$ skeleton configurations $(g_i)_{i \leq m} \in G$ sampled over time. In practice, we found that breakdance sequences provide interesting variability in poses, with an associated set of 10 to 20 geodesics offering a range of motion large enough to allow generalization to other motion. Each configuration is a pair $(c, p) \in G = SE(3) \times SO(3)^n$, where $n \in \mathbb{N}$ is the number of joints, $c$ is the rigid transformation describing the configuration of the root bone, and $p$ contains the relative orientation of each joint with respect to its parent.

We will denote the pose space of relative joint orientations by $P = SO(3)^n$. We apply PGA on the pose samples $(p_i)_{i \leq m}$ to obtain the intrinsic mean $\mu \in P$, and a set of $k \in \mathbb{N}$ geodesic directions $V_j \in \mathfrak{p}$ describing the pose submanifold. With this decomposition, the new pose DOFs are $k$ scalars $(\alpha_j)_{j \leq k} \in \mathbb{R}$ describing coordinates over the $k$ principal geodesics. Practical computation of the intrinsic mean $\mu$ and geodesic directions $V_j$ are detailed in [TWC*09].

We define the *reduced* pose parametrization as a smooth function $r : \mathbb{R}^k \to P$ that reconstructs a pose given $k$ geodesic coordinates, $\alpha = (\alpha_j)_{j \leq k}$. Two mappings may be chosen for pose reconstruction, known as the canonical coordinates of the *first* and *second* kind [MLS94]. In practice, both have provided similar results for our purpose. We thus restrict to the first one:

$$r(\alpha) = \mu \exp\left(\sum_{j=1}^{k} \alpha_j . V_j\right)$$

We obtain the complete character forward kinematics, including the root DOFs, by composing $r$ with the usual forward kinematics for articulated rigid bodies. We summarize this mapping as the following function:

$$f : SO(3) \times \mathbb{R}^3 \times \mathbb{R}^k \to SE(3)^{n+1} \tag{3}$$

where the configuration space $G := SO(3) \times \mathbb{R}^3 \times \mathbb{R}^k$ is a Lie group describing respectively the absolute orientation, absolute position and reduced pose coordinates of the character. $f$ computes the absolute configuration of each of the $k+1$ bones of the skeleton.

The differential of $f$ is obtained by the chain rule. Note, however, that the derivative of the exponential does not assume a simple formula as it does for the real scalar case, due to the non-commutativity of the rotation group. See [MLS94] for appropriate derivation.

## 4. Principal Geodesic Dynamics

Having described how to use a PGA-based reduced pose model in a kinematic animation context, we now move on to the physically-based animation of a character, using this reduced model as generalized coordinates. In this section we motivate and describe a time-stepping scheme for physically animating a virtual character parameterized using PGA. We derive a geometric integrator based on [KCD09]. Finally, we propose an approximated, explicit time-integrator, for use in a real-time simulation.

The use of reduced coordinates is natural in our context since we already have the reduced coordinates pose mapping as the PGA pose parametrization. In this case the constraint forces are implicit, and no constraint drift can occur by design. However, this formalism imposes the derivation of dedicated equations of motion, and notably to compute the reduced mass tensor and Coriolis forces according to the reduced coordinates mapping. In this context, the low-dimension of the PGA parametrization will be an advantage since it will keep the computational cost reasonable while still enforcing natural poses.

### 4.1. Geometric Integrator Derivation

We now derive the discrete equations of motion for a PGA-parametrized virtual character. For this, we use the geometric, variational integrator methodology presented in

[KCD09]. This work addressed the case of a single rigid body and consequently made the assumption of the left-invariance[†] of the Lagrangian. Instead, we derive equations for the general case of multiple articulated rigid bodies, where such an assumption does not hold. Unless stated otherwise, all the tangent maps and vectors used in the following are expressed in *body* coordinates, as opposed to *spatial* coordinates [MLS94].

Let $f : G \to SE(3)^{n+1}$ be the forward kinematics as defined in Equation (3), and $J(g)$ its Jacobian matrix. We assume that the inertia tensors for the $n+1$ bones composing the skeleton are given as a positive definite, block-diagonal matrix $\widehat{M}$ of dimension $6(n+1)$. The kinetic energy $T$ is then defined as:

$$T(g, v) = \frac{1}{2} v^T J(g)^T \; \widehat{M} \; J(g) v$$

where $g \in G$ is the configuration, and the velocity $v \in \mathfrak{g}$ belongs to the Lie algebra. The above expression defines the generalized mass matrix, or inertia tensor as:

$$M(g) = J(g)^T \; \widehat{M} \; J(g)$$

The potential energy $V(g)$ is simply the gravitation potential.

**Variational Integrator** Assuming a Lagrangian $\mathcal{L} = T - V$ is given, the Hamilton equations of motion can be derived from the Hamilton-Pontryagin (HP) principle [KYT*06]. By properly discretizing the HP principle through a *quadrature*, one can obtain a symplectic integrator of arbitrary accuracy (called variational integrator) with excellent momentum and energy conservation properties. In the case of Lie groups, [KCD09] propose a first-order accurate quadrature of the HP principle in terms of a *group difference* map $\tau : G \to \mathfrak{g}$:

$$\delta \int_0^T \left( \mathcal{L}(g, v) + p^T (\dot{g} - v) \right) \, dt \quad \approx \tag{4}$$

$$\delta \sum_{k=0}^{N} h \mathcal{L}(g_k, v_{k+1}) + p_{k+1}^T \left[ \tau(g_k^{-1} g_{k+1}) - h \, v_{k+1} \right] = 0$$

where the momentum $p^T \in \mathfrak{g}^*$ belongs to the Lie coalgebra, and $h$ is the time step. In practice, we used the logarithm as the group difference map. The stationary conditions described by the above discretized HP principle result in a symplectic update rule for $g, v$ and $p$, summarized in the following non-linear system (see Annexe A for more details):

$$\frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) \, d\tau_{k+1} = p_k^T d\tau_k \, Ad_{d_k} + h \frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1}) \tag{5a}$$

$$g_{k+1} = g_k \tau^{-1}(h \, v_{k+1}) \tag{5b}$$

$$p_{k+1}^T = \frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) \tag{5c}$$

where $d_{k+1} = g_k^{-1} g_{k+1}$, $d\tau_k := d\tau(d_k^{-1})$ and $Ad$ is the adjoint[‡] representation of $G$. Forcing can be incorporated through an extended variational principle [KCD09]. If $f_k$ is an external force applied at the $k^{th}$ time step, the integrated force $h \, f_k$ is added to the velocity update (5a):

$$\frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) \; d\tau_{k+1} = p_k^T d\tau_k \, Ad_{d_k} \quad + \tag{6}$$
$$h \left( \frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1}) + f_k^T \right)$$

Equations (5b) and (5c) are left unchanged.

### 4.2. Approximations for Real-Time Simulation

Two non-linear expressions in $v_{k+1}$ complicate the integration update in equation (6). We propose two approximations to turn the update into a linear system, at the expense of symplecticity. Our experimental results show however that angular momentum is nearly conserved, which is not the case when using a non-Lie group integrator.

#### 4.2.1. Quadratic Forces

The presence of the quadratic forces [MLS94] $\frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1})$, containing Coriolis and centrifugal forces, is problematic since this term is quadratic in $v_{k+1}$. To avoid this issue, we approximate it by using the velocities from the previous time step, thus making it completely explicit:

$$\frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1}) \approx \frac{\partial \mathcal{L}}{\partial g}(g_k, v_k)$$

In our simulator, we compute this term using central finite differences.

#### 4.2.2. Difference Map Derivative

The second cause of non-linearity in the above variational update is the presence of $d\tau_{k+1}$ in the left-hand side of Equation (6), and this term depends non-linearly on $v_{k+1}$. As above, we approximate this term by reusing the velocity from the previous time step, which corresponds to $d_{k+1} \approx d_k$. Under these approximations, the velocity update becomes the following *linear* system:

$$\frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) = \Bigg[ p_k^T \; d\tau_k \, Ad_{d_k} \quad + \tag{7}$$
$$h \left( \frac{\partial \mathcal{L}}{\partial g}(g_k, v_k) + f_k^T \right) \Bigg] d\tau_k^{-1}$$

Note that if a more accurate *predictor* is available for $v_{k+1}$, the two approximations presented above can be adapted in a straightforward manner.

[†] *i.e.* constant in body coordinates

[‡] For a Lie group $G$, the adjoint $Ad_x : \mathfrak{g} \to \mathfrak{g}$ maps body coordinates to spatial coordinates corresponding to a tangent vector at $x \in G$.

### 4.3. Evaluation

We now evaluate the proposed integrator. To show the advantages of Lie group integration, we compare our integrator with one of the following form:

$$\frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) = p_k^T + h\left(\frac{\partial \mathcal{L}}{\partial g}(g_k, v_k) + f_k^T\right) \qquad (8)$$

Equation (8) is a naive adaptation of the Euclidean symplectic Euler integrator to the Lie group case, with explicit quadratic forces for the sake of comparison. The extra terms $d\tau_k Ad_{d_k}$ and $d\tau_k^{-1}$ in Equation (7) can be seen as appropriate change of frames for taking the Lie group *curvature* into account when summing external forces and previous momentum. A graphical interpretation of this can be found in [KCD09].

We compare the behavior of these two integrators on the following scenario: the character is in a gravity-less environment, we apply a short impulse on its right hand, along the positive Z axis. We choose a quite large time-step for an explicit integrator: $h = 0.1s$.

The accompanying video shows the trajectory of the Angular Momentum (AM) vector at the center of mass position. The Euclidean integrator clearly shows instability that our integrator does not exhibit, as seen in Figure 3. Another experiment is performed with a fall under gravity onto an infinite inclined slope. Again, the conservation of momentum appeared to be preserved by our integrator while some artifacts arose for the Euclidean integrator.

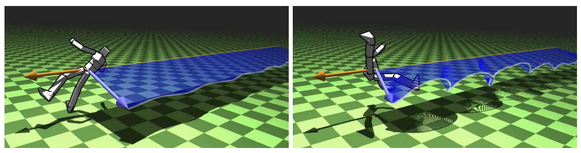
**Figure 3:** *The Angular Momentum* (blue) *is displayed after a short lateral impulse in a gravity-less environment. Our Lie group integrator* (left) *features a much better AM conservation than a non Lie-group one* (right).

## 5. Control Formulation

Now that a physically-based animation model has been derived, we turn to the problem of motion control.

### 5.1. Quadratic Programming

We describe a character control framework in the same spirit of [LMH10], but at the velocity/impulse level instead of force/acceleration. Equation (7) performs an explicit velocity update as a linear system, which can be summarized in a compact velocity-impulse formulation $Mv = p$. If we incorporate contact impulses, non-penetration and complementarity constraints into this system, we obtain the Karush-Kuhn-Tucker (KKT) conditions of a convex Quadratic Program (QP). We now formulate motion control by altering this QP.

The constrained dynamics of a character can be summarized as the following QP:

$$v_{k+1} = \underset{Av \geq b}{\operatorname{argmin}} \quad \frac{1}{2}v^T M v - p^T v \qquad (9)$$

where $v$ is the velocity coordinates, $p$ is the net momentum (*i.e.* the left-hand side in Equation (7)), $M$ is the mass tensor, and $A, b$ describe unilateral constraint geometry. We modify the original quadratic form, using a *control* quadratic form $\frac{1}{2}v^T Qv + c^T v$, in order to obtain a different behavior:

$$v_{k+1} = \underset{Av \geq b}{\operatorname{argmin}} \quad \underbrace{\frac{1}{2}v^T M v - p^T v}_{\text{dynamics}} + \underbrace{\frac{1}{2}v^T Q v + c^T v}_{\text{control}} \qquad (10)$$

The control form may represent any *soft* kinematic constraint one wishes to satisfy on velocities, expressed as a quadratic form. We will see how this form is constructed in the next paragraphs. To enrich the user control, a simple interpolation parameter can be employed for the two quadratic terms to balance between a purely physical simulation, or better follow the kinematic constraints at the expense of physical realism.

One should note that modifying the original quadratic form in the above way effectively amounts to adding implicit forces of the form $Qv + c$, as can be seen on the corresponding KKT conditions. If the control form involves the root DOFs of the character, the associated forces *directly* act on the character root.

We now describe how to construct the control form in terms of control features.

### 5.2. Features Tracking Formulation

In practice, the control form is the aggregation of quadratic error terms, each one controlling a specific velocity-dependent *feature*. Let $F \simeq \mathbb{R}^d$ be an abstract *feature space* of dimension $d$. We define a *feature* $\gamma$ as an affine map $\gamma$:

$$\gamma : \mathfrak{g} \to F$$
$$\gamma(v) = \Gamma v - \widetilde{\gamma}$$

We call $\Gamma \in \mathcal{M}_{d,n}$ the feature *matrix*, and $\widetilde{\gamma} \in F$ the feature *desired velocity*. Given a set of $m \in \mathbb{N}$ features $(\gamma_i)_{i \leq m}$, we build a quadratic form by simply summing all the feature squared norms:

$$\frac{1}{2}v^T Qv + c^T v := \frac{1}{2}\sum_{i=1}^{m} ||\gamma_i(v)||^2$$

More precisely, $Q = \sum_i \Gamma_i^T \Gamma_i$, $c = -\sum_i \Gamma_i^T \widetilde{\gamma}_i$, and unneeded constant terms are dropped. Features can be given more importance in the final control form by *weighting* them accordingly. Such a weight is implicit in the definition of each feature, in the above formula.

We now illustrate the above formulation on a simple IK example. Let a function $e : G \to \mathbb{R}^3$ describe the error between an end-effector position and its desired position. The error for the current configuration $g_k$ is $e_k := e(g_k)$. We obtain an estimation of $e_{k+1}$ based on a first-order approximation at the current time step:

$$e_{k+1} \approx e_k + h\, J_{e_k}\, v$$

where $J_{e_k}$ is the Jacobian matrix of $e$ at $g_k$. We would like $e_{k+1}$ to be as small as possible, thus we try to enforce:

$$J_{e_k}\, v = -\frac{e_k}{h}$$

The corresponding control feature is described by the pair $(J_{e_k}, e_k/h)$. In this example, we ask the error to be zero at the next time step, which corresponds to a *proportional* gain of $\phi_p = 1/h$. Smaller gains can also be provided, as well as *derivative* gains, in order to obtain a general Proportional-Derivative (PD) model for control features:

$$J_{e_k}\, v = -\phi_p\, e_k - \phi_d\, \dot{e_k} \tag{11}$$

where $\dot{e_k} = J_{e_k} v_k$ is the error derivative for the current time step. As one would expect, the derivative gain $\phi_d$ tends to damp oscillations around the control target, at the expense of convergence speed. In the case where the error function is defined on a Lie group (*e.g.* when controlling orientation), the Lie group PD control described in [BM95] can be used instead. An example of the IK control feature is shown in Figure 4.

We now present some control results obtained using our system.

## 6. Examples of Controllers

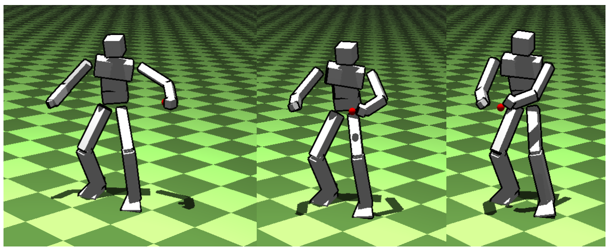

**Figure 4:** *Simple IK control feature. The desired position* (red) *is interactively manipulated by the user while the character maintains balance.*

### 6.1. Looking at Targets

Due to the full-body correlations captured by the PGA pose model, we have found the head rotational motion to be sometimes too important and thus penalizing for visual quality. We thus implemented the following target tracking feature in order to gain an intuitive control on the head orientation.

The absolute head orientation is defined by a mapping $q : G \to SO(3)$. We compute the desired orientation as a direct orthogonal basis of $\mathbb{R}^3$ as follows:

- The first basis vector $b_1$ is the normalized view vector $w \in \mathbb{R}^3$, constructed as the difference between the view target and the head center
- The second basis vector $b_2$ is the projection of the world up vector $u \in \mathbb{R}^3$ onto the plane orthogonal to $b_1$, normalized
- The third basis vector $b_3$ is obtained by taking the cross-product of the two first vectors, finalizing the basis

Such construction is obviously ill-defined when the view and up vectors coincide. Should this situation happen, we simply skip this feature in the final control form.

Once the desired orientation $q^\star \in SO(3)$ is computed, we define an error function $e(g) = \log({q^\star}^{-1} q(g)) \in \mathbb{R}^3$ and obtain the corresponding control feature as described in 5.2. Figure 5 shows an example use of this feature control.

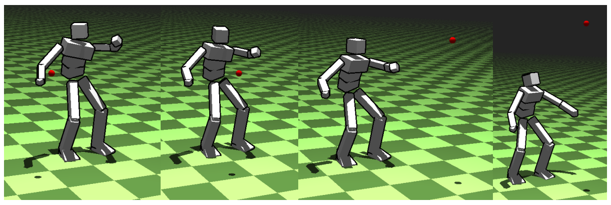

**Figure 5:** *An example of head orientation control: the character is looking at the red target.*

### 6.2. Balance Control

Let us now turn to balance-related feature control. As shown in [MZS09], the linear and angular momenta are two high-level quantities of particular interest in this context.

#### 6.2.1. Center of Mass (CoM)

The CoM position is given by a function $c : G \to \mathbb{R}^3$ defined as:

$$c(g) = \sum_{i=1}^{n+1} m_i\, c_i(g)$$

where $n+1 \in \mathbb{N}$ is the number of rigid bodies in the skeleton, $c_i$ computes the position of the CoM for the i$^{\text{th}}$ body, and $m_i$ is the mass for the i$^{\text{th}}$ body.

Given a target position for $c$, an error function is easily derived from the above definition. Again, we obtain the corresponding velocity feature as described in 5.2.

### 6.2.2. Angular Momentum (AM)

In a feature-based control framework, it is possible to *directly* control the AM [LMH10], as opposed to an indirect AM control through the Zero Momentum Point [MZS09]. The AM with respect to the CoM $c$ is defined as:

$$a = \sum_{i=1}^{n+1} R_i\, \mathcal{I}_i\, \omega_i \;+\; \sum_{i=1}^{n+1} m_i (c_i - c) \times \dot{c}_i \quad \in \mathbb{R}^3$$

where $R_i \in SO(3)$ is the absolute orientation of the ith body, $\mathcal{I}_i$ is the body-fixed inertia tensor of the ith bone, $\omega_i$ is the ith body-fixed angular velocity. The AM is linear in the rigid body velocities, which are themselves linear in the generalized velocities. Thus, the AM may be rewritten as a linear function of the generalized velocities:

$$a = H\, v$$

Thus, $H$ is used as the AM feature matrix. We set the desired AM value to zero in order to prevent the character from undergoing destabilizing rotational motion.

Figures 6 and 7 compare the resulting behaviors when the AM control is turned off and on.

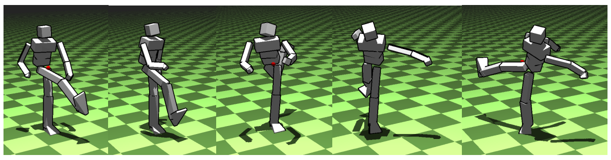
**Figure 6:** *One foot balance, without AM control: the character makes broad moves and quickly falls down. The red marker shows the desired CoM position.*

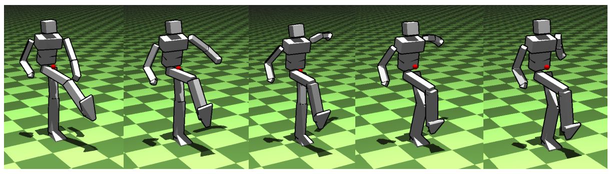
**Figure 7:** *One foot balance, with AM control: the character adjusts his arms and moves in a balanced way. The red marker shows the desired CoM position.*

### 6.2.3. Support Center

Instead of fully controlling the CoM position, it is usually sufficient to keep its projection on the ground *inside* the support polygon [JYL09, LMH10]. To do so, we record the position of the character contacts with the ground at each time step. We define the center of support $s \in \mathbb{R}^3$ as the mean of these contact points (*cf.* Figure 8). A control feature is added so that the CoM projection on the ground matches the center of support.

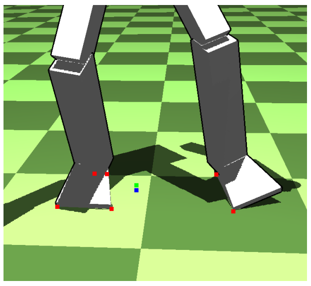
**Figure 8:** *The contact points* (red) *are used to define the support center* (blue) *as their mean. The CoM projection* (green) *is controlled by setting its desired value to the blue point.*

### 6.2.4. Results

Our balance controller uses the following feature set:

- Head and CoM projections should be at the support center,
- Feet should stay on the ground,
- Angular momentum should be minimal,
- CoM height.

We found the head projection criteria to be a simple yet effective way to maintain the character upright. Figure 1 and Figure 9 show examples of one-foot balance control. The accompanying video shows extended examples of interaction, with on-line change of support foot controlled by the user. This behavior is simply obtained by biasing the CoM and head projection targets toward the supporting foot position. The overall posture changes (standing, crouching, jumps) of the character are implemented through change in target height of the CoM.

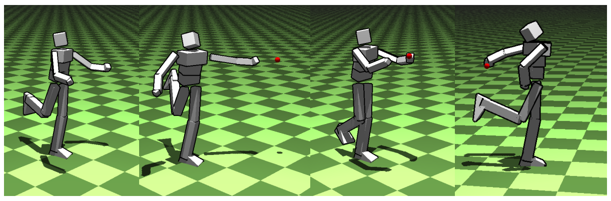
**Figure 9:** *One foot balance, while tracking an IK target.*

## 7. Discussion and Limitations

**Efficiency** As expected, the dimension reduction offered by the PGA allows to improve efficiency, since all the examples we proposed run in full real-time, at around 100Hz on a quad-core machine. We believe these performances to be improvable by engineering a better optimized code.

**Joint Limits** Though not described here for clarity, we have also added handling of joint limits. Our approach is based on a sampling of values observed in the learning joint orientation data. A Minimum Volume Enclosing Ellipsoid is fitted to these data as a compact, analytical description of the allowed joint space. Unilateral constraints preventing the joint orientation from leaving this ellipsoid were incorporated in the control framework described in section 5. A conceptually similar, but more complex approach can be found in [HUH02].

**Robustness** Though the balance controllers we present performed generally well under user interaction, summing control terms in an unprioritized way will necessarily be subject to robustness problem in the case of a user insisting too much. We did not implement yet a prioritized optimization strategy as found in [LMH10], though it could have largely benefited from our reduced dimension strategy. This kind of framework produces much more robust controllers than simply summing objective terms, as the core balance objectives can not be perturbed by user IK requests. This could also improve the robustness of our controllers to impulsive perturbations.

**Generalization** Principal geodesic DOFs are completely dependent on the choice of the motion capture sequence used for learning. It thus questions the generalization capacity of such an approach. In our experiments, we mainly used the breakdance sequences for learning as shown in the accompanying video. It shows good properties for the interaction scenario we provide. The computation of PGA is extremely fast (less than a second in the example shown). The learning phase is not a limitation and different learning data could be tested on the fly if the resulting geodesics seem not appropriated to the type of motion edition wished by the user. More complex behavior such as locomotion is beyond the intended scope of this paper. As shown by numerous previous studies, simulation of locomotion requires careful controllers (typically finite state machines are needed). Our goal here was to firstly show that features-based controllers could be implemented in our Principal Geodesic Dynamics framework. The development of more complex controllers implementing locomotion is a natural extension of our work.

## 8. Conclusion

A Quadratic Programming control approach, exploiting a PGA reduced pose parametrization, has been presented. This approach allows to easily add small dynamic effects to the kinematic manipulation of a virtual character. We have experienced the system to be stable enough even when the contacts scenario is not known in advance. While such a control framework presented above can produce good visual results, it is not totally physically correct since the character is *externally* actuated through the features tracking. A more satisfying, but also more difficult approach, is to only use the *internal* actuators of the character to achieve a given task expressed using velocity features. Doing this is much more difficult from a theoretical point of view, as the true task-space control problem under unilateral constraints is non-convex, and thus much harder to solve. Some algorithms have been proposed for solving such problems, such as a Sequential Quadratic Program. We did not try these methods yet, looking for real-time solutions in the first place. However, although these iterative algorithms are computationally-expensive, our reduced pose model may finally provide a low enough dimension reduction to make their real-time use possible.

## Annexe A: Discrete Geometric Integrator

Let $d_{k+1} = g_k^{-1} g_{k+1}$. We now derive the stationary conditions for momentum, velocity and position in Equation (4).

**Momentum** Taking variations in $p$ gives:

$$\sum_{k=0}^{N} \delta p_{k+1}^T \left(\tau(d_{k+1}) - h\, v_{k+1}\right) = 0$$

The *fundamental lemma of the calculus of variations* implies that for all $k \leq N$:

$$\delta p: \quad \tau(d_{k+1}) = h\, v_{k+1} \in \mathfrak{g} \tag{12}$$

**Velocity** Taking variations in $v$ gives:

$$\sum_{k=0}^{N} \left( \frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) - p_{k+1}^T \right) \delta v_{k+1} = 0$$

The same argument implies that for all $k \leq N$:

$$\delta v: \quad \frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1}) = p_{k+1}^T \in \mathfrak{g}^* \tag{13}$$

**Position** Let us first remark that $\tau(x^{-1}) = -\tau(x)$, and let $d\tau_k := d\tau(d_k^{-1})$. Taking variations in $g$ with fixed end-points ($\delta g_0 = \delta g_{N+1} = 0$) gives:

$$\sum_{0}^{N} h \frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1})\, \delta g_k - p_{k+1}^T d\tau_{k+1} \delta d_{k+1}^{-1} = 0 \tag{14}$$

Noting that $\delta d_{k+1}^{-1} = -Ad_{d_{k+1}} \delta g_{k+1} + \delta g_k$, we split the sum by grouping terms in $\delta g_k$ and $\delta_{g_{k+1}}$ together. After renumbering and exploiting the fixed endpoints property, we end up with the following equation for all $k \leq N$:

$$\delta g: \quad \frac{\partial \mathcal{L}}{\partial v}(g_k, v_{k+1})\, d\tau_{k+1} = p_k^T d\tau_k\, Ad_{d_k} + h.\frac{\partial \mathcal{L}}{\partial g}(g_k, v_{k+1}) \tag{15}$$

## Acknowledgements

The authors wish to thank the reviewers for their helpful feedback. This work was partly funded by the French ANR project SoHuSim and the French ANR project Morpho.

## References

[AdSP07] Abe Y., da Silva M., Popović J.: Multiobjective control with frictional contacts. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2007), Eurographics Association, pp. 249–258. 3

[BH00] BRAND M., HERTZMANN A.: Style machines. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 183–192. 2

[BM95] BULLO F., MURRAY R. M.: Proportional derivative (pd) control on the euclidean group. In *In European Control Conference* (1995), pp. 1091–1097. 7

[FLPJ04] FLETCHER P. T., LU C., PIZER S. M., JOSHI S. C.: Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging 23*, 8 (2004), 995. 3, 4

[GMHP04] GROCHOW K., MARTIN S. L., HERTZMANN A., POPOVIĆ Z.: Style-based inverse kinematics. *ACM Trans. Graph. 23*, 3 (2004), 522–531. 2

[HUH02] HERDA L., URTASUN R., HANSON A.: Automatic determination of shoulder joint limits using quaternion field boundaries. *In Proceedings of the 5th International Conference on Automatic Face and Gesture Recognition 2002* (2002), 95–100. 9

[JBP06] JAMES D. L., BARBIČ J., PAI D. K.: Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. *ACM Trans. Graph. 25*, 3 (July 2006), 987–995. 2

[JL11] JAIN S., LIU C. K.: Modal-space control for articulated characters. *ACM Trans. Graph. 30*, 5 (Oct. 2011), 118:1–118:12. 2

[JYL09] JAIN S., YE Y., LIU C. K.: Optimization-based interactive motion synthesis. *ACM Trans. Graph. 28*, 1 (February 2009), 10–1. 3, 8

[KCD09] KOBILAROV M., CRANE K., DESBRUN M.: Lie group integrators for animation and control of vehicles. *ACM Trans. Graph. 28*, 2 (May 2009), 16–1. 4, 5, 6

[KRFC09] KRY P., REVÉRET L., FAURE F., CANI M.-P.: Modal locomotion: animating virtual characters with natural vibrations. *Comput. Graph. Forum 28*, 2 (avr 2009), 289–298. Special Issue: Eurographics 2009. 2

[KYT*06] KHAREVYCH L., YANG W., TONG Y., KANSO E., MARSDEN J. E., SCHRÖDER P., DESBRUN M.: Geometric, variational integrators for computer animation. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2006), SCA '06, Eurographics Association, pp. 43–51. 5

[LMH10] LASA M. D., MORDATCH I., HERTZMANN A.: Feature-based locomotion controllers. *ACM Transactions on Graphics 29number* (2010). 3, 6, 8, 9

[MK05] MUKAI T., KURIYAMA S.: Geostatistical motion interpolation. *ACM Trans. Graph. 24*, 3 (July 2005), 1062–1070. 2

[MLH10] MORDATCH I., LASA M. D., HERTZMANN A.: Robust physics-based locomotion using low-dimensional planning. *ACM Transactions on Graphics 29*, 3 (2010). 3

[MLS94] MURRAY R. M., LI Z., SASTRY S. S.: *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., Boca Raton, FL, USA, 1994. 3, 4, 5

[Moa02] MOAKHER M.: Means and averaging in the group of rotations. *SIAM J. Matrix Anal. Appl. 24*, 1 (2002), 1–16. 4

[MZS09] MACCHIETTO A., ZORDAN V., SHELTON C. R.: Momentum control for balance. *ACM Trans. Graph. 28*, 3 (2009), 1–8. 3, 7, 8

[NCNV*12] NUNES R. F., CAVALCANTE-NETO J. B., VIDAL C. A., KRY P. G., ZORDAN V. B.: Using natural vibrations to guide control for locomotion. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2012), I3D '12, ACM, pp. 87–94. 2

[Pen06] PENNEC X.: Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision 25*, 1 (2006), 127. 4

[PW89] PENTLAND A., WILLIAMS J.: Good vibrations: modal dynamics for graphics and animation. In *Proceedings of the 16th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1989), SIGGRAPH '89, ACM, pp. 215–222. 2

[RCB98] ROSE C., COHEN M. F., BODENHEIMER B.: Verbs and adverbs: Multidimensional motion interpolation. *IEEE Comput. Graph. Appl. 18*, 5 (1998), 32–40. 2

[SHP04] SAFONOVA A., HODGINS J. K., POLLARD N. S.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 514–521. 2

[SLHN10] SOMMER S., LAUZE F., HAUBERG S., NIELSEN M.: Manifold valued statistics, exact principal, geodesic analysis and the effect of linear, approximations. In *Proceedings of the 11th European conference on Computer vision: Part VI* (Berlin, Heidelberg, 2010), ECCV'10, Springer-Verlag, pp. 43–56. 4

[TLP06] TREUILLE A., LEWIS A., POPOVIĆ Z.: Model reduction for real-time fluids. *ACM Transactions on Graphics 25*, 3 (July 2006), 826–834. 2

[TWC*09] TOURNIER M., WU X., COURTY N., ARNAUD E., REVERET L.: Motion compression using principal geodesic analysis. *Computer Graphics Forum (EUROGRAPHICS'09)* (Mar. 2009). 3, 4

[WMC11] WEI X., MIN J., CHAI J.: Physically valid statistical models for human motion generation. *ACM Trans. Graph. 30*, 3 (May 2011), 19:1–19:10. 2

[WST09] WICKE M., STANTON M., TREUILLE A.: Modular bases for fluid dynamics. *ACM Transactions on Graphics* (July 2009). 2

[YL08] YE Y., LIU C. K.: Animating responsive characters with dynamic constraints in near-unactuated coordinates. In *ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), SIGGRAPH Asia '08, ACM, pp. 112–1. 3