EUROGRAPHICS 2007 / D. Cohen-Or and P. Slavík
(Guest Editors)

Volume 26 (2007), Number 3

Soft Articulated Characters with Fast Contact Handling

Nico Galoppo[1], Miguel A. Otaduy[2], Serhat Tekin[1], Markus Gross[2] and Ming C. Lin[1]

[1] Department of Computer Science, UNC Chapel Hill, USA [2] Computer Graphics Laboratory, ETH Zurich, Switzerland

Abstract

Fast contact handling of soft articulated characters is a computationally challenging problem, in part due to complex interplay between skeletal and surface deformation. We present a fast, novel algorithm based on a layered representation for articulated bodies that enables physically-plausible simulation of animated characters with a high-resolution deformable skin in real time. Our algorithm gracefully captures the dynamic skeleton-skin interplay through a novel formulation of elastic deformation in the pose space of the skinned surface. The algorithm also overcomes the computational challenges by robustly decoupling skeleton and skin computations using careful approximations of Schur complements, and efficiently performing collision queries by exploiting the layered representation. With this approach, we can simultaneously handle large contact areas, produce rich surface deformations, and capture the collision response of a character's skeleton.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation I.3.5 [Computer Graphics]: Physically Based Modeling

1. Introduction

Believable animation of articulated characters is essential to realistic virtual environments, computer games, and other interactive applications. Often the movement of characters is driven by motion capture data. However, it is important to simulate the response of characters to collisions, as well as secondary effects, such as skin wrinkles or surface deformation, thus enhancing the realism of the character animation. But, these effects are usually challenging to achieve in real time. Various techniques have been proposed for performing skeleton-driven deformations [MTLT88, LCF00, MG03, KZ05], some with physically based skin deformation [CHP89, GTT89, CGC*02]. However, none of these approaches addresses the problem of efficiently handling global and local response to contact constraints. Surface contact is difficult to model using traditional skinning techniques because the combination of bone transforms and blend weights completely determine the resulting (deformed) shape.

Our approach for simulating soft characters with contact constraints constitutes perhaps the first unified framework for real-time modeling of skeletal deformations, surface deformation due to contact, and their interplay on object surfaces with thousands of degrees of freedom. Our algorithm can efficiently handle large contacts, automatically produce rich surface deformations, and effectively capture the skeletal response of the characters due to collisions.

Our algorithm builds upon a conceptually simple and commonly used *layered representation* for soft characters in computer animation, which is essentially an integration of articulated body dynamics and skinning with displacement corrections. Such a representation obviously cannot capture general global deformations, but it is well suited for representing skeletal and surface deformations. One of the challenges for modeling soft articulated characters that has not been well investigated previously is the interplay of skeletal motion and surface contact and the resulting two-way coupling effects. Another major issue is the enforcement of contact constraints on soft articulated bodies with many degrees of freedom. Our algorithm takes advantage of a fast, approximate, image-space collision detection algorithm for deformable characters whose surface is computed by displacements from (weighted) rigid bones, and it overcomes the computational bottlenecks due to contacts with the following key results:

- New formulation of elastic deformation in pose space, which is related to skin displacement corrections [KJP02, JT05] and FEM approaches in a floating frame of refer-

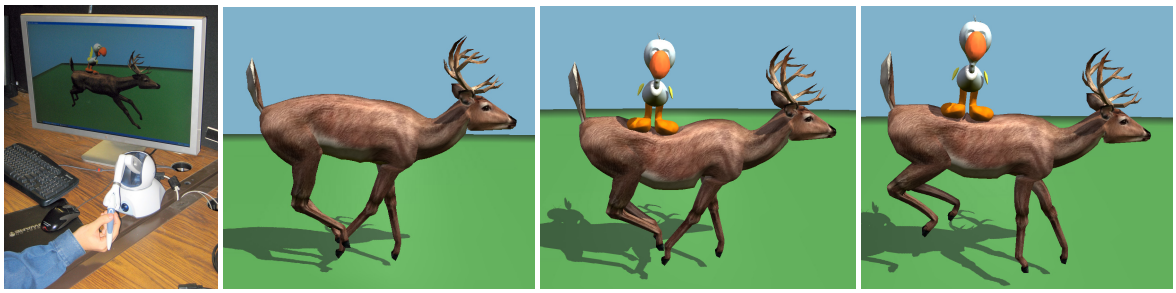

Figure 1: Interactive Deformation of an Articulated Deer. *The deer, consisting of* 34 *bones and* 2755 *deformable surface vertices is being deformed interactively (almost* 10 *fps on average) by a rigid bird model. The interplay between small-scale contact deformations and the skeletal contact response is successfully captured.*

ence [TW88], augmented with a joint stiffness term to model pose-dependent deformations. With this formulation, the motion equations derived from Lagrangian mechanics naturally produce the desired interplay between skin and skeleton.

- Efficient and scalable computation of articulated-body dynamics with contact constraints and skin deformations, with a cost of $O(m+k+n)$ in practice, where m is the number of contacts, k the number of bones, and n the number of surface nodes. Our method is based on the decoupling of skin and skeleton computations in otherwise coupled implicit equations, through careful and robust approximation of Schur complements.

We continue with a discussion of prior work in Section 2, and a description of our layered representation for soft characters in Section 3. In Section 4, we present the dynamic formulation of articulated body dynamics with skin deformations and contact constraints. Section 5 describes our efficient solution to the constrained dynamics problem, and Section 6 highlights the performance of our complete approach on complex benchmarks. We conclude with a discussion of possible future research directions.

2. Related Work

Modeling of deformable articulated characters is a problem that has been investigated using data driven, example based, or physically based approaches. The latter are best suited for simulating collision response, and here we focus the discussion mainly on methods that tackle the simulation of the skeleton and/or surface deformations.

Several linear-time algorithms exist for simulating articulated skeletons without closed loops, either with articulated body inertias [Fea87] or with Lagrange multipliers [Bar96]. We formulate joint constraints with Lagrange multipliers as this framework can also be used to formulate contact constraints naturally. Recently, a few researchers have shown how to handle both unilateral and bilateral constraints. For example, Cline and Pai [CP03] emphasize handling rigid body contact constraints using post-stabilization, whereas Erleben [Erl04] combines joint constraints, joint limits, and joint motors with rigid contact constraints in a velocity-based linear complementarity formulation with shock propagation. Weinstein et al. [WTF06] propose an iterative solution for handling joint and contact constraints for rigid, articulated bodies in a single framework. Accurate contact handling for an articulated body with k bones and m contacts has $O(km)$ complexity [Bar96]. We show how to reduce this complexity to a linear dependence on k and m in Section 5. We should note that physically based simulation of an articulated skeleton can also be combined with motion capture data to generate plausible blending of motion capture segments [ZMCF05].

Given the animation of the skeleton, it is possible to model the deformation of the skin surface by linear blending of bone transformations, the so-called *skeletal-subspace deformation* (SSD) [MTLT88]. Different methods have been proposed to address the problems of linear blending by using example-based deformations [LCF00], adding eigenbases of deformations in pose space [KJP02], inserting additional joints tuned from examples [MG03], or employing blending of transformations instead of weights [KZ05], among others. Recent techniques have extended skinning to mesh deformations [JT05], motion capture data without a predefined skeleton [PH06], or interactive models [DSP06].

Simple skinning fully defines surface positions based on the skeletal configuration but cannot capture the reaction of the character to collisions. In the simulation of human characters, accurate anatomy-based representations (e.g., [DCKY02]) can be used for modeling material properties. These representations, however, are computationally expensive, and our goal is to develop methods that can interactively capture surface deformation effects and global deformation in a plausible way with less focus on internal behavior. Our approach can be classified under the category of layered deformable models [CHP89, GTT89, TT93, Gas98], which overlay layers of deformable material on top of an articulated skeleton that drives the motion. Skeletal deformations can also be used to impose constraints on a control lattice for FEM simulation of dynamic deformations [CGC*02, GW05]. Recently, Capell et al. [CBC*05]

extended their framework to include rigging force fields, self-collision handling, and linearization of deformations in pose space. In contrast to earlier work that performed a simulation of the complete volume of the model, we focus the computational effort on the surface and the skeleton. In addition, we provide efficient solutions that capture the interplay between surface deformation effects and skeletal motion due to collisions. Some earlier deformation techniques have also focused the computations on the surface, such as matrix condensation [BNC96, JP02], the boundary element method [JP99], quasi-rigid body models [SK03, PPG04], or deformation textures [GOM*06], but none of them handles global deformations due to impact dynamics for an articulated model. Finally, our approach is also related to methods that extract a local reference frame for applying a linear elasticity model [TW88, MT92, MDM*02], in particular the pose-space method of Capell et al. [CBC*05]

3. Overview of the Soft Character Model

In this section we describe our formulation of deformations in the pose space of an articulated character, and discuss the FEM discretization of the deformation field.

3.1. Pose Space Deformation

In the skeletal-subspace deformation model with k bones, the deformed position $\mathbf{x}$ of a material point is defined based on the position $\mathbf{u}$ in pose space $\mathsf{T}_{o,i}$ and bone transformations T_i as

$$\mathbf{x} = \sum_{i=1}^{k} w_i \mathsf{T}_i \mathbf{u}_i = \left(\sum_{i=1}^{k} w_i \mathsf{T}_i \mathsf{T}_{o,i}^{-1} \right) \mathbf{u}. \qquad (1)$$

We choose to express deformation and elastic energy in pose space (also known as the rest configuration of the mesh) before applying the skin transformations (see Figure 2), as proposed before for geometric deformation and displacement corrections [JT05, KJP02, LCF00]. Pose space offers a local coordinate frame on which we can measure elastic energy using a linear strain tensor without suffering from geometric non-linearities [MDM*02].

The deformed position in pose space $\mathbf{u}$ can be decomposed into a constant undeformed component $\mathbf{u}_o$ and an elastic skin displacement $\mathbf{u}_s$, hence $\mathbf{u} = \mathbf{u}_o + \mathbf{u}_s$. We choose the bone transforms to be rigid transforms, and then we can write $\mathbf{u}_i = \mathsf{T}_{o,i}^{-1}\mathbf{u} = \mathbf{c}_{o,i} + \mathbf{R}_{o,i}\mathbf{u}$ and $\mathsf{T}_i\mathbf{u}_i = \mathbf{c}_i + \mathbf{R}_i\mathbf{u}_i$, with $\mathbf{c}_i$, $\mathbf{c}_{o,i}$ displacements and $\mathbf{R}_i$, $\mathbf{R}_{o,i}$ rotation matrices. The constant transformations $\mathbf{c}_{o,i}$ and $\mathbf{R}_{o,i}$ transform world-space surface positions in rest state to each bone's reference system. We assume that the blend weights w_i obey the affine constraint $\sum_i w_i = 1$.

3.2. Discretization and Meshing

In our reduced model representation, the degrees of freedom (DoFs) are determined by the DoFs of the bone transforms T

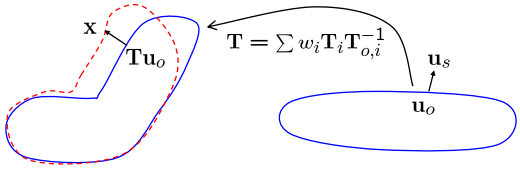

Figure 2: Pose space deformation. *Elastic deformations* $\mathbf{u}_s$ *of the skin are defined in rest pose space*

and the DoFs of the deformable layer. We discretize the deformation field $\mathbf{u}_s$ in the deformable layer using linear FEM . The deformation field $\mathbf{u}_s$ can then be approximated by n discrete node values accumulated in a vector $\mathbf{q}_s \in \mathbb{R}^{3n}$ through the (position-dependent) shape matrix $\mathbf{S}$ and expressed compactly as $\mathbf{u}_s = \mathbf{S}\mathbf{q}_s$.

An advantage of our method is that we can generate dynamic models from skinned meshes that have been created with popular 3D authoring software. A volumetric mesh of the deformable layer can be generated with any method that preserves the original outer surface vertices, either by defining two enclosing surfaces [HGB06] or by generating a layer of tetrahedral elements inwards from the outer surface [EDS05]. The mesh blend weights can simply be reused for the physical model as defined in Eqn. (1).

By replacing the deformation field in Eqn. (1) with its discretized version, we obtain:

$$\mathbf{x} = \sum_{i=1}^{k} w_i(\mathbf{c}_i + \mathbf{R}_i(\mathbf{c}_{o,i} + \mathbf{R}_{o,i}(\mathbf{u}_o + \mathbf{S}\mathbf{q}_s))). \qquad (2)$$

The DoFs of our model can be packed together in the generalized state vector $\mathbf{q} = \begin{bmatrix} \mathbf{q}_b \\ \mathbf{q}_s \end{bmatrix}$, where $\mathbf{q}_b = \begin{bmatrix} \mathbf{c}_1^T & \theta_1^T & \dots & \mathbf{c}_k^T & \theta_k^T \end{bmatrix}^T \in \mathbb{R}^{7k}$ for k bones. We chose quaternions to represent the orientations θ.

The velocity state vector $\mathbf{v}$ follows from the time differentiation of (2). As shown in Appendix A, the world-frame velocity $\dot{\mathbf{x}}$ of a material point can be approximated as $\dot{\mathbf{x}} = \mathbf{L}_W\mathbf{v}$, for a velocity state vector $\mathbf{v} = \begin{bmatrix} \mathbf{v}_b \\ \mathbf{v}_s \end{bmatrix}$, with $\mathbf{v}_b = \begin{bmatrix} \dot{\mathbf{c}}_1^T & \omega_1^T & \dots & \dot{\mathbf{c}}_k^T & \omega_k^T \end{bmatrix}^T \in \mathbb{R}^{6k}$ and angular bone velocities ω expressed in the bone's local frame. The velocity state vector and generalized coordinate vector are related by $\mathbf{v} = \mathbf{P}\dot{\mathbf{q}}$ and $\dot{\mathbf{q}} = \mathbf{P}^+\mathbf{v}$, with $\mathbf{P}$ and $\mathbf{P}^+$ matrices that transform angular velocities ω to time derivatives of quaternions in $\dot{\mathbf{q}}$. Note that our discretized deformation model (2) is identical to the deformation model in [GOM*06] for the case of a single bone ($k = 1$).

4. Soft Characters with Contact Constraints

In this section we formulate the constrained dynamic simulation problem for soft characters. We model both joint and contact constraints with the method of Lagrange multipli-

ers, and we use implicit Backward Euler integration with linearization of forces.

4.1. Coupled Layered Dynamics

We formulate the dynamic motion equations of our soft articulated characters using Lagrangian continuum mechanics [GPS02, Sha89]. Using linear elasticity theory and linear FEM and by formulating the displacement field of the soft layer in pose space, we can regard elastic forces as linear with respect to the displacements $\mathbf{u}_s$, and as invariant to the rigid bone transformations T in Eqn. (1) [Sha89, TW88].

The elastic energy given by pose-space displacements and linear elasticity yields the usual sparse block $\mathbf{K}_s$ of the stiffness matrix $\mathbf{K}$ that affects only DoFs of the soft skin layer $\mathbf{q}_s$, not the skeleton. However, note that, as shown in Section 5.5, the skeletal response of surface contact forces is naturally captured by our model. On the other hand, our formulation of pose space strain does not model pose-dependent strain energy as in e.g., elbow bending. We partially capture this effect in the skeleton dynamics by adding a joint stiffness term between connected bones in the skeleton. This leads to off-diagonal non-zero blocks in the skeleton stiffness block $\mathbf{K}_b$ (see Appendix B).

The kinetic energy depends on both skeleton and skin velocities, and it captures the interplay of articulated motion and skin deformation. Our linearized deformation model in pose space bears similarity with the one of Capell et al. [CBC*05], but we effectively capture inertial forces by directly considering pose space deformations in the Lagrangian formulation instead of using corotational methods [MDM*02].

The inertia matrix $\mathbf{M} = \int \rho \mathbf{L}_W^T \mathbf{L}_W dV$ [Sha89] (See Appendix A) has a structure $\mathbf{M} = \begin{pmatrix} \mathbf{M}_b & \mathbf{M}_{bs} \\ \mathbf{M}_{bs}^T & \mathbf{M}_s \end{pmatrix}$, with $\mathbf{M}_b \in \mathbb{R}^{6k\times 6k}$ the inertia of bones and $\mathbf{M}_s \in \mathbb{R}^{3n\times 3n}$ the inertia of the skinned surface. Due to skinning blend-weights, $\mathbf{M}_b$ computed from the full Lagrangian would present off-diagonal blocks. However, the inertial coupling between bones is dominated by joint constraints, hence we compute a block-diagonal $\mathbf{M}_b$, where for each bone we compute the inertia by associating approximate link geometry (See Figure 4). Note that this approximation requires that the bone coordinate frames are located at the center of mass of the approximate link geometries. The inertia of the skin, $\mathbf{M}_s$, can be tuned to produce effects such as the wobbling shown in the accompanying video. The dense bands $\mathbf{M}_{bs}$ and $\mathbf{M}_{bs}^T$ are key for capturing the effect of surface contact forces on bone motion through inertial coupling.

We incorporate damping $\mathbf{D}$, generalized external forces $\mathbf{Q}$, and a quadratic velocity vector $\mathbf{Q}_v$ [Sha89] that represents the inertial effects of centripetal and Coriolis forces on the bones. By assembling all terms, we obtain the following set of ordinary differential equations (ODEs):

$$\begin{cases} \mathbf{M}\dot{\mathbf{v}} = \mathbf{Q} + \mathbf{Q}_v - \mathbf{K}\mathbf{q} - \mathbf{D}\mathbf{v} = \mathbf{F} + \mathbf{J}_\mu^T \mu + \mathbf{J}_\lambda^T \lambda, \\ \dot{\mathbf{q}} = \mathbf{P}^+ \mathbf{v}. \end{cases} \tag{3}$$

We explicitly separate joint constraint forces $\mathbf{J}_\mu^T \mu$ and contact forces $\mathbf{J}_\lambda^T \lambda$ from other forces $\mathbf{F}$. In the discrete formulation of joint and contact forces (see Sections 4.2 and 4.3), the Lagrange multipliers μ and λ will include the time discretization Δt, and can be regarded as impulses.

We have discretized the motion equations using implicit backward Euler with a first order approximation of forces, as this allows for stable and responsive contact response [GOM*06]. The discretized equations have the form $\tilde{\mathbf{M}}\Delta\mathbf{v} = \Delta t \tilde{\mathbf{F}}$, with discrete-time mass matrix $\tilde{\mathbf{M}}$ and discrete-time force vector $\tilde{\mathbf{F}}$ defined as

$$\tilde{\mathbf{M}} = \mathbf{M} - \Delta t \frac{\partial \mathbf{F}}{\partial \mathbf{v}} - \Delta t^2 \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \mathbf{P}^+ \text{ and } \tilde{\mathbf{F}} = \mathbf{F} + \Delta t \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \mathbf{P}^+. \tag{4}$$

Given the separation of forces in Eqn. (3), and as performed in a similar manner by others [Erl04, CW05], we decompose the dynamic update into three steps: (i) computation of collision free velocities $\mathbf{v}^- = \mathbf{v}(t-\Delta t) + \Delta\mathbf{v}$ from the old velocities, (ii) collision detection and identification of contact constraints, and (iii) computation of collision response $\delta\mathbf{v}$ that yields constrained velocities $\mathbf{v}(t) = \mathbf{v}^- + \delta\mathbf{v}$. To ensure enforcement of constraints on positions as well, we apply a final correction step that projects (possibly) penetrating vertices to the constraint surfaces.

4.2. Joint Constraints

We use the method of Lagrange multipliers to compute joint constraint forces [Bar96]. It is important to observe that the joint constraints do not influence the skin coordinates. Hence, $\mathbf{J}_\mu$ is of the form $\mathbf{J}_\mu = \begin{bmatrix} \mathbf{J}_j & \mathbf{0} \end{bmatrix}$, with $\mathbf{J}_j$ a sparse $(c \times 6k)$ block matrix, k the number of bones, and c the total number of DoFs of the joints. The non-zero $(d \times 6)$ blocks in $\mathbf{J}_j$ are defined by each pair of bones connected with a d-DoF joint. Given the joint Jacobians, we can then define constraints on the collision-free velocities as follows (we have also added a stabilization term $-\alpha\mathbf{g}(\mathbf{q}_b)$ to avoid drift):

$$\mathbf{J}_\mu \mathbf{v}^- = -\alpha\mathbf{g}(\mathbf{q}_b) \quad \Rightarrow \quad -\mathbf{J}_j \Delta\mathbf{v}_b = \mathbf{J}_j \mathbf{v}_b + \alpha\mathbf{g}(\mathbf{q}_b). \tag{5}$$

By combining the discrete motion equations and the joint constraints (5), we can arrange the collision-free velocity update of the articulated skeleton in a large linear system:

$$\begin{pmatrix} \tilde{\mathbf{M}}_b & \tilde{\mathbf{M}}_{bs} & -\mathbf{J}_j^T \\ \tilde{\mathbf{M}}_{bs}^T & \tilde{\mathbf{M}}_s & \mathbf{0} \\ -\mathbf{J}_j & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{v}_b \\ \Delta\mathbf{v}_s \\ \mu \end{pmatrix} = \begin{pmatrix} \Delta t \tilde{\mathbf{F}}_b \\ \Delta t \tilde{\mathbf{F}}_s \\ \mathbf{b}_\mu \end{pmatrix}, \tag{6}$$

with $\mathbf{b}_\mu = \mathbf{J}_j \mathbf{v}_b + \alpha\mathbf{g}(\mathbf{q}_b)$.

The system above is sparse, symmetric, and indefinite, with

a rather dense band $\tilde{\mathbf{M}}_{bs}$, due to the inertial coupling between the skin and the skeleton. The size of this band ($O(kn)$ with k bones and n surface nodes) can be regarded as a lower bound on the cost for solving the system with direct solvers [BBK05], and the indefiniteness of the system suggests slow convergence of iterative solvers. In Section 5.2 we propose a solution combining matrix condensation and an approximation of skin forces that decouples the system and reduces the bilinear complexity.

4.3. Contact Constraints

We apply collision response by formulating velocity constraints on colliding surface nodes and solving them through the method of Lagrange multipliers. Instead of simply applying an impulse to the colliding nodes, we formulate the constraints on the implicit motion equations, which guarantees that collision response effectively acts on the skeletal motion as well [GOM*06]. Our collision detection algorithm as described in Section 5.3 identifies one contact constraint with normal $\mathbf{n}$ for each colliding surface node. Given the pre-impact velocity $\dot{\mathbf{x}}_i^-$ of the colliding node, we look for the node contact response $\delta\dot{\mathbf{x}}_i$ by imposing a velocity constraint using the kinematic relationships (Eqn. (17)):

$$\mathbf{n}^T\left(\dot{\mathbf{x}}_i^- + \delta\dot{\mathbf{x}}_i\right) = \mathbf{n}^T\mathbf{L}_W^i\left(\mathbf{v}^- + \delta\mathbf{v}\right) = 0, \qquad (7)$$

where $\mathbf{L}_W^i$ represents the position-dependent matrix $\mathbf{L}_W$ evaluated at node i. We can easily incorporate moving constraints, friction, and constraint correction. Moving constraints are handled by a velocity offset in the contact constraints in Eqn. (7). Similar to the approach of Bridson et al. [BFA02], friction and constraint correction can be handled for each node separately as a post-process to $\delta\mathbf{v}_s$, and to the resulting post-collision state $\mathbf{q}_s$ respectively.

The velocity constraints can be stacked together in $\mathbf{J}_\lambda = \begin{bmatrix} \mathbf{J}_b & \mathbf{J}_s \end{bmatrix} \in \mathbb{R}^{m\times(6k+3n)}$, where m is the number of colliding surface nodes, k is the number of bones, and n is the total number of surface nodes. The constraint equation is then:

$$\mathbf{J}_\lambda\left(\mathbf{v}^- + \delta\mathbf{v}\right) = 0 \Rightarrow -\mathbf{J}_b\delta\mathbf{v}_b - \mathbf{J}_s\delta\mathbf{v}_s = \mathbf{J}_b\mathbf{v}_b^- + \mathbf{J}_s\mathbf{v}_s^-. \qquad (8)$$

With constraints formulated through Lagrange multipliers, the complete system of equations for collision response is:

$$\begin{pmatrix} \tilde{\mathbf{M}}_b & \tilde{\mathbf{M}}_{bs} & -\mathbf{J}_j^T & -\mathbf{J}_b^T \\ \tilde{\mathbf{M}}_{bs}^T & \tilde{\mathbf{M}}_s & 0 & -\mathbf{J}_s^T \\ -\mathbf{J}_j & 0 & 0 & 0 \\ -\mathbf{J}_b & -\mathbf{J}_s & 0 & 0 \end{pmatrix} \begin{pmatrix} \delta\mathbf{v}_b \\ \delta\mathbf{v}_s \\ \mu \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mathbf{b}_\mu^- \\ \mathbf{b}_\lambda \end{pmatrix}, \qquad (9)$$

$$\text{with } \mathbf{b}_\mu^- = \mathbf{J}_j\mathbf{v}_b^- + \alpha\mathbf{g}(\mathbf{q}_b^-) \text{ and } \mathbf{b}_\lambda = \mathbf{J}_b\mathbf{v}_b^- + \mathbf{J}_s\mathbf{v}_s^-.$$

The system above can be regarded as an augmented version of Eqn. (6), with the addition of contact constraints. A direct application of a method such as constraint anticipation [Bar96] would yield a computational cost $O(mkn)$ at best, since the complete system described in Eqn. (6) should be solved for each contact. However, we propose a solution with a practical $O(m+k+n)$ cost in the next section.

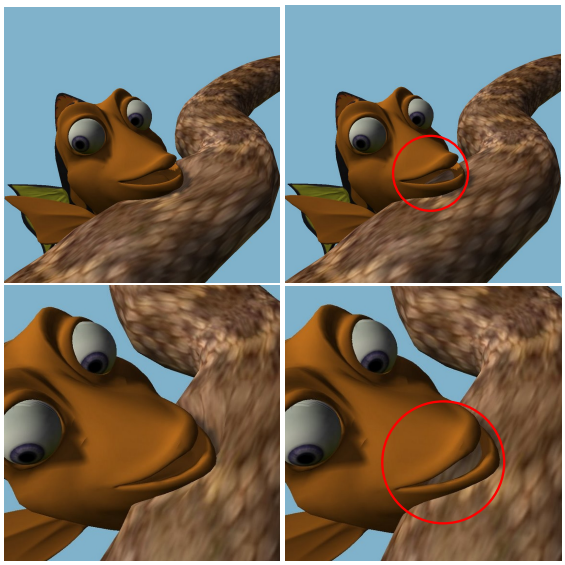

Figure 3: Contact Constraints. *Left column: The fish touches the body of the snake, creating global response and skin deformations. Right column: We turn off local skin deformations to show the importance of handling both global and surface response. Notice the highlighted interpenetrations, clearly visible through the fish's mouth. The comparison is also shown in the supplementary video footage.*

5. Condensed Solution of Constraints

Two common constraints need to be resolved in the dynamics simulation of soft articulated characters, namely joint and contact constraints. One of our key contributions in this work is to reduce the best-case $O(mkn)$ complexity of the full solution to contact constraints, while preserving physically plausible global and local deformation effects. We achieve this by combining Schur complement computation [GV96] (also referred to as matrix condensation [BNC96]), and careful approximations of implicit discretization. We first present the condensation of skeleton dynamics, which allows for $O(k+n)$ update of collision-free dynamics in practice. Then we present the condensation of contact constraints and anticipation of skeleton response, which allow for $O(m+k+n)$ update of contact-consistent dynamics in practice.

5.1. Condensed Skeleton Dynamics

Joint constraints act only on bone coordinates, not on skin coordinates, and we exploit this observation to split the velocity computation in Eqn. (6) and Eqn. (9), solving first for bone velocities (while accounting for forces on the skin). Computing the Schur complement of the skin inertia $\tilde{\mathbf{M}}_s$ yields a *condensed skeleton inertia* $\tilde{\mathbf{M}}_{\text{cond}} = \tilde{\mathbf{M}}_b - \tilde{\mathbf{M}}_{bs}\tilde{\mathbf{M}}_s^{-1}\tilde{\mathbf{M}}_{bs}^T \in \mathbb{R}^{6k\times 6k}$. Unfortunately, computing $\tilde{\mathbf{M}}_{\text{cond}}$ requires solving $6k$ linear systems of size n, and the resulting matrix is dense.

Instead we compute an approximate condensed matrix

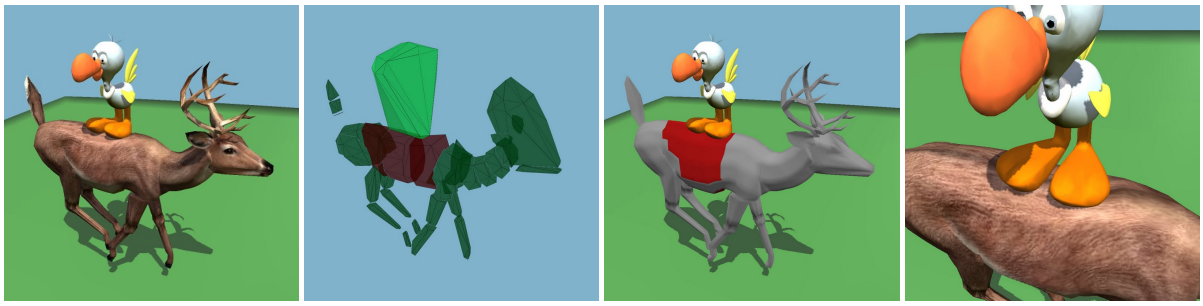

Figure 4: Layered Representation and Collision Detection. *From left to right: Contact between the bird and the deer, with skin deformations on the back of the deer; Proxies used for hierarchical pruning of collision queriess, with potentially colliding proxies of the deer highlighted in red; Triangles influenced by the potentially colliding bones (in red) are the only ones passed to our image-based collision detection algorithm; The resulting detail around the contact area.*

$\hat{\mathbf{M}}_{\text{cond}} = \tilde{\mathbf{M}}_b - \tilde{\mathbf{M}}_{bs}\hat{\mathbf{M}}_s^{-1}\tilde{\mathbf{M}}_{bs}^T$, where $\hat{\mathbf{M}}_s^{-1}$ is a fast approximate inverse of $\tilde{\mathbf{M}}_s$ that accounts only for block-diagonal terms. In this way the computation of $\hat{\mathbf{M}}_{\text{cond}}$ has an $O(n+k)$ complexity. The approximation $\hat{\mathbf{M}}_s$ amounts to discarding off-diagonal blocks of $\mathbf{K}$ (i.e., the Jacobians of elastic forces among skin nodes) in the implicit computation of velocities. Note that this approximation does not jeopardize the fulfillment of joint or contact constraints; it simply yields velocities that differ slightly from those of the full solution with Eqn. (4). Moreover, we employ the full inverse $\tilde{\mathbf{M}}_s^{-1}$ in the computation of collision-free skin velocities in Eqn. (11). We have quantified the error $\|\hat{\mathbf{M}}_{\text{cond}} - \tilde{\mathbf{M}}_{\text{cond}}\|/\|\tilde{\mathbf{M}}_{\text{cond}}\|$ (using the spectral norm), and it is below 10% in our simulations, with some variation depending on the average number of bone influences per vertex.

5.2. Solving Collision-Free Velocities

Applying condensed skeleton dynamics, we can rewrite the constrained system for bone velocities in Eqn. (6) as:

$$\begin{pmatrix} \hat{\mathbf{M}}_{\text{cond}} & -\mathbf{J}_j^T \\ -\mathbf{J}_j & \mathbf{0} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{v}_b \\ \mu \end{pmatrix} = \begin{pmatrix} \mathbf{b}_b \\ \mathbf{b}_\mu \end{pmatrix}, \tag{10}$$

$$\text{with } \mathbf{b}_b = \Delta t \left(\tilde{\mathbf{F}}_b - \tilde{\mathbf{M}}_{bs}\tilde{\mathbf{M}}_s^{-1}\tilde{\mathbf{F}}_s \right) \text{ and } \mathbf{b}_\mu = \mathbf{J}_j\mathbf{v}_b + \alpha\mathbf{g}(\mathbf{q}_b).$$

The structure of Eqn. (10) is practically the same as one would obtain when solving a regular articulated body with implicit integration of joint stiffness. Equation (10) can be solved in $O(k+c)$ time, with c the number of DoFs constrained by the joints, for articulated structures without loops [Fea87, Bar96]. But $\hat{\mathbf{M}}_{\text{cond}}$ presents off-diagonal nonzero blocks for pairs of connected bones (due to joint stiffness), or pairs of bones who contribute to a common skin patch in the linear-blend skinning scheme. Hence we have opted for a more general indefinite symmetric sparse system solver with fill-reducing reordering [SG06]. Since the sparsity pattern of Eqn. (10) is fixed, the reordering and analysis (or symbolic factorization) can be precomputed. This approach makes the runtime cost of solving this system linear in the number of bones and joints in our simulation. Therefore, it was not a bottleneck, as discussed in Section 6.

After solving Eqn. (10) and computing collision-free bone velocities $\Delta\mathbf{v}_b$, we solve for skin velocities $\Delta\mathbf{v}_s$ in Eqn. (6):

$$\tilde{\mathbf{M}}_s\Delta\mathbf{v}_s = \Delta t\tilde{\mathbf{F}}_s - \tilde{\mathbf{M}}_{bs}^T\Delta\mathbf{v}_b. \tag{11}$$

The matrix $\tilde{\mathbf{M}}_s$ is constant and symmetric positive-definite. We also compute a fill-reducing sparse factorization of $\tilde{\mathbf{M}}_s$ once for the entire simulation [SG06]. As discussed in Section 6, the performance of the solver is linear in our experiments, and better than a diagonally preconditioned conjugate gradient method. In summary, by approximate condensation of skeleton dynamics, we are able to reduce the brute-force $O(nk)$ complexity to $O(k+n)$ in practice.

5.3. Hiearchical Pruning and Collision Queries

One of the essential components to efficiently solve contact constraints is a fast collision detection module. We adopt a fast image-based algorithm that exploits the layered representation of our soft characters. We execute collision detection in two steps: (i) identifying contact patches with object-space techniques using low-resolution proxies, and (ii) checking for high-resolution skin surface collisions and collecting colliding skin vertices using image-space techniques with the aid of graphics hardware. Our method shares the two-step approach of others used for rigid bodies [OJSL04] or rigid bodies with a deformable surface layer [GOM*06]. But unlike these methods, our collision query algorithm also performs hierarchical pruning to eliminate large portions of the objects from collision queries, by exploiting the skeletal nature of the deformation. The worst-case cost of our collision detection is $O(n)$ for a pair of tested objects with n surface nodes; the actual cost depends on the size of the contact area.

For the object-space collision detection step, we assign to each bone a low-resolution proxy, as shown in Figure 4. Specifically, for the i^{th} bone we construct as preprocessing a low polygon-count approximate convex hull of the set of skin vertices $\{v\}$ with blend weight $w_i \neq 0$. Typically, our convex proxies have a few tens of vertices. At runtime we use the maximum skin displacement $\mathbf{u}_s$ of all vertices asso-

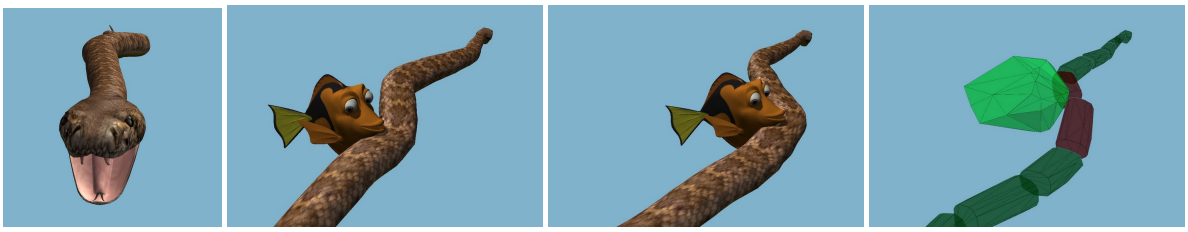

Figure 5: Skeletal Deformations of a Soft Snake. *Simulation sequence with a fish touching the snake, showing the global deformation of the snake. The last image shows the proxies for collision detection.*

ciated to a bone for computing a conservative bound of the proxy. We identify potentially colliding bones using a fast collision detection algorithm for convex objects [EL00] that identifies low-resolution proxies that are within a user specified tolerance distance of each other.

For the image-space detection of actual contacts, we use the approach of [GOM*06]. But, our method can handle self-collisions between surface patches of the same articulated body. However, the second step in our collision detection algorithm may report false positives for adjacent bone surface patches, especially for adjacent polygons in the mesh. This case is handled by a post-processing step to check for exact collision between adjacent polygons in object space.

5.4. Condensed Contact Constraints

When a collision has been detected, we apply the same principle of condensation of skeleton dynamics as described in Section 5.1 to solve the collision reponse Equations (9). However, this condensation is not sufficient to separate the computation of bone and skin response, since contact constraints act on the bones as well as on the skin, as shown in Eqn. (8). Our solution is to compute *condensed contact constraints* $\mathbf{J}_{\text{cond}}$ with the approximated $\hat{\mathbf{M}}_s^{-1}$ from Section 5.1:

$$\mathbf{J}_{\text{cond}} = \mathbf{J}_b - \mathbf{J}_s \hat{\mathbf{M}}_s^{-1} \tilde{\mathbf{M}}_{bs}^T.$$

Algebraic manipulation of Eqn. (8) and Eqn. (9) yields the *condensed constraint equations*:

$$\mathbf{M}_\lambda \lambda + \mathbf{J}_{\text{cond}} \delta\mathbf{v}_b + \mathbf{b}_\lambda = 0, \tag{12}$$

$$\text{with } \mathbf{M}_\lambda = \mathbf{J}_s \hat{\mathbf{M}}_s^{-1} \mathbf{J}_s^T. \tag{13}$$

Here, λ is the Lagrange multiplier vector that defines the contact impulses. In order to split the equations, we propose the *anticipation of skeleton response*, i.e., to single out $\lambda = -\mathbf{M}_\lambda^{-1} \mathbf{J}_{\text{cond}} \delta\mathbf{v}_b - \mathbf{M}_\lambda^{-1} \mathbf{b}_\lambda$. Other existing methods for solving multibody dynamics with joint and contact constraints propose the anticipation of contact constraints (i.e. first singling out skeleton response) and then solving for the contact impulses [Bar96]. However, in our setting we exploit the use of equality contact constraints, the fact that each colliding surface node yields one constraint, and the approximation of skin force Jacobians. These together make the matrix $\mathbf{M}_\lambda$ diagonal and trivial to invert. Thereby, we significantly reduce the overall computational cost of expensive contact constraint anticipation.

5.5. Solving Collision Response

By application of condensed skeleton dynamics, condensed contact constraints, and anticipation of skeleton response, we obtain the following system of equations for computing *contact-consistent* articulated dynamics:

$$\begin{pmatrix} \mathbf{M}_b^* & -\mathbf{J}_j^T \\ -\mathbf{J}_j & \mathbf{0} \end{pmatrix} \begin{pmatrix} \delta\mathbf{v}_b \\ \mu \end{pmatrix} = \begin{pmatrix} \mathbf{b}_b^* \\ \mathbf{b}_\mu^- \end{pmatrix}, \tag{14}$$

$$\text{with } \mathbf{M}_b^* = \hat{\mathbf{M}}_{\text{cond}} + \mathbf{J}_{\text{cond}}^T \mathbf{M}_\lambda^{-1} \mathbf{J}_{\text{cond}} \text{ and } \mathbf{b}_b^* = -\mathbf{J}_{\text{cond}}^T \mathbf{M}_\lambda^{-1} \mathbf{b}_\lambda.$$

The matrix $\mathbf{M}_b^*$ has exactly the same structure as $\hat{\mathbf{M}}_{\text{cond}}$; therefore, we use the same solver (and precomputed symbolic factorization) for articulated dynamics as discussed in Section 5.2. Remarkably, with our method the addition of a soft skin to the articulated character enables a solution of collision response on the skeleton with the same cost as the collision-free solution.

Once the skeletal response $\delta\mathbf{v}_b$ is computed, we obtain the collision impulse λ as discussed in the Section 5.4, and finally we solve for the skin response in Eqn. (9) as:

$$\delta\mathbf{v}_s = \hat{\mathbf{M}}_s^{-1} \left(\mathbf{J}_s^T \lambda - \tilde{\mathbf{M}}_{bs}^T \delta\mathbf{v}_b \right). \tag{15}$$

The approximation of skin force Jacobians largely simplifies the solution of skin response, and we have not encountered associated instabilities in the simulations. Our observation is that the use of full Jacobians in the collision-free update as described in Eqn. (11) guarantees stability, while the coupled response computed on the skeleton ensures the global reaction to collisions.

To summarize, the solution of bone velocities in Eqn. (10) and Eqn. (14) presents a cost $O(k)$, the solution of skin velocities in Eqn. (11) and Eqn. (15) presents a cost $O(n)$, and the condensation and anticipation steps are sparse matrix multiplications with overall cost $O(m+k+n)$. Altogether, the solution of constrained dynamics with our soft articulated characters presents a final cost $O(m+k+n)$, as also observed in our experiments.

5.6. Run-time Algorithm

Here we outline the different substeps performed by our algorithm in each simulation step. We decompose each dynamic simulation step into a collision-free update step, followed by contact response.

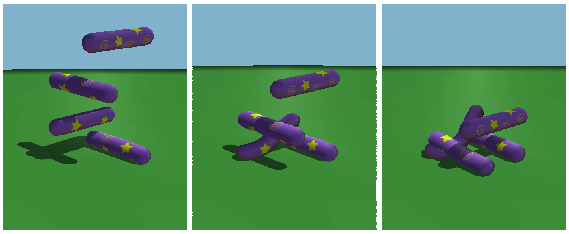

Figure 6: Contact between Deformable Tubes with Moving Constraints. *The tubes consist of 3 links each and collide with each other in tangled configurations. Our algorithm can handle such situations seamlessly with a combination of local deformations and bone motion at 20 fps.*

1. Compute free-motion bone velocities (Eqn. (10)).
2. Compute collision-free skin velocities $\mathbf{v}_s^-$ (Eqn. (11)).
3. Update collision-free positions.
4. Execute collision detection (See Section 5.3).
5. Formulate condensed constraint equations (Eqn. (12)).
6. Formulate contact-consistent articulated dynamics (Eqn. (14)) to compute skeletal contact response.
7. Compute contact impulses λ (Section 5.4) and solve for skin response (Eqn. (15)).
8. Update contact-consistent positions.
9. Collision detection and collision-free position reprojection.

6. Results

We have tested our algorithm on a variety of benchmarks, using the soft articulated characters listed in Table 1. All the benchmarks were simulated on a 3.4 GHz Pentium-4 processor PC with an NVidia GeForce 7800GTX graphics card. We have performed several experiments with a set of bendable tubes (see Figure 6) for validating the behavior with different material stiffness, friction values, number of bones, and moving constraints. The deer model in Figure 1 was driven by a pre-recorded animation and by applying additional control forces on its bones.

Table 1 also shows a breakdown of the average timings per frame, highlighting the time for collision-free dynamics update, collision detection, and collision response. The last two columns show the average total time per frame, for (1) non-colliding situations and (2) colliding situations. Note that, for the benchmark of the snake (16 bones and 3 102 skin nodes, shown in Figure 5), the simulation runs at 7 fps with collisions, and 10 fps when there are no collisions. For the benchmark of the deer (34 bones and 2 755 skin nodes), the simulation is also interactive (as shown in Figure 1) in the range of $6-9$ fps. For one tube with 5 bones, the simulation runs at 43 fps even with large colliding areas. The example shown in Figure 6 runs at $15-20$ fps.

For the deer model, we have performed the simulation with different skin resolutions. As shown in the table, the simulation cost varies linearly w.r.t. the resolution of the skin n. Furthermore, notice how, for the two lower-resolution skins, the cost for the collision-free velocity update of the bones remains almost unchanged and independent of the skin resolution. This data shows that the cost at those resolutions is dominated by the number of bones and, more importantly, that the cost for solving bone and skin velocities is not bilinear $O(kn)$ with our algorithm. Similarly, we have observed that the number of collisions for the mid-resolution deer varies from 1 to 41, while the time for collision response on the bones remains practically unchanged during the simulation. This data again shows that the cost at low resolutions is dominated by the number of bones and is not bilinear $O(km)$ w.r.t. the number of contacts. Combining the observations for collision-free updates and collision response, we can conclude that the simulations have the runtime complexity of $O(m+k+n)$ in practice. It is interesting to notice that, for the deer model with the highest resolution mesh, the dominating cost is the initialization of matrices for the computations, not the solution of the constrained systems.

We have also compared the performance of the sparse system solver [SG06] with preconditioned conjugate gradient descent (PCG) to solve (11), on the deer model with 2 755 surface nodes. We used a diagonal preconditioner consisting of the diagonal part of $\tilde{\mathbf{M}}_s$. We found that PCG is 4 times slower for solving the collision-free update of the skin, even after reaching 100 iterations without fully converging.

As demonstrated in the experiments, our method for simulating soft articulated characters handles contact constraints interactively while producing rich deformations on the skin. Due to its layered representation, it cannot be directly compared with methods that model global deformations using a volumetric representation. From the family of FEM-based methods, the one by Müller et al. [MDM*02] is perhaps the closest relative to ours, as it also uses implicit integration and linear elasticity evaluated in a floating reference frame. In comparison with Müller's, our method is obviously restricted to skeletal global deformations, not arbitrary ones, and the elastic energy only accounts for skin-layer deformations, not deformations in the whole volume of the object. However, our method offers considerable benefits for fast collision handling. Due to our fast solver presented in Section 5, we can efficiently compute global collision response (i.e., response of the skeleton) using matrix condensation. Furthermore, we can handle both local skin and global skeletal response with approximate implicit integration stably and robustly. Müller's method (and others) cannot exploit the decomposition of the deformation, and the constraint-based simulation would require the use of the full implicit system, in order to robustly compute global response. Perhaps for this reason, methods such as Müller's are often combined with penalty-based collision response, not constraint-based, with the associated problem of object interpenetration as shown in Fig. 3.

Model	Nodes n	Bones k	Coll. $\max(m)$	Collision-Free Update Setup	Bones	Skin	Coll. Det.	Collision Response Setup	Bones	Skin	Total Time[1]	Total Time[2]
Deer	1 748	34	13	19.7	34.0	17.9	6.1	11.1	15.8	7.7	74.8	123.8
Deer	2 755	34	41	39.3	36.6	29.6	10.8	11.3	14.7	9.9	114.1	160.4
Deer	8 408	34	162	127.0	64.9	96.2	41.1	24.5	17.8	33.9	310.0	449.7
Snake	3 102	16	28	38.9	18.1	36.5	13.6	6.2	2.2	12.9	106.5	140.6
Tube	292	2	73	2.5	1.6	1.9	3.9	0.8	0.2	0.8	9.5	13.8
Tube	292	5	74	3.7	2.7	2.1	7.7	1.5	0.6	1.0	14.6	23.3

Table 1: Statistics of our Benchmarks. *The soft characters for the benchmarks are a deer model with three different skin resolutions, a snake model, and tubes with different numbers of bones. All timings (in msec.) are averages over the course of a simulation. The last two columns indicate the average time per frame in (1) non-colliding and (2) colliding situations.*

7. Summary and Future Work

We have presented a novel method for simulating deformable characters with an articulated skeleton that allows fast handling of complex contact scenarios and plausible, coupled global and local deformations. Our method models characters as skinned articulated bodies with a layered representation, and measures elastic deformations in pose space. Central to the efficient simulation of contact-induced deformations is an implicit constraint-based collision handling approach that exploits the layered representation to enable efficient, approximate yet robust matrix condensation. We are able to achieve interactive simulation rates on models with challenging contact configurations.

Our deformation model expressing strain in pose space does not capture pose-dependent strain near joints. We have approximated pose-dependent strain energy with a user-tunable joint stiffness, but our model could be further extended with data-driven approaches [LCF00] to add e.g., bulging effects due to elbow flexion. This extension would integrate well with our linear deformation model in pose space. Another possible extension for handling more complex volumetric deformations would be to adopt a deformation model based on a control lattice [CGC*02], instead of a purely skeletal approach. This would require an extension of our contact handling algorithm for efficiently applying collision response on both the control lattice and the skeleton. Similarly, it is worth exploring the addition of our efficient contact-induced deformations to non-skeletal skinning approaches [JT05].

We are exploring ways to extend our simulation algorithm to handle inequality constraints, as this would allow modeling more accurate contact forces and joint limits. We are also investigating a parallel solution for hardware-accelerated implementation, which would increase the performance and thus enable more complex scenes with many deforming characters at interactive rates.

Acknowledgements This work is supported in part by ARO, NSF, RDECOM, Intel Corporation and the NCCR CoMe of the Swiss NSF. We'd like to thank the GAMMA group in Chapel Hill and the CGL in Zurich for their support, in particular Mario Botsch and Bob Sumner for help with sparse solvers and mesh processing; Matthias Mueller (Ageia), Kenny Erleben (DIKU) and John Ratcliff (Simutronics) for help with tetrahedralization and fruitful discussion regarding multi-body dynamics. The authors would also like to thank the anonymous reviewers for their critical feedback to improve the paper.

References

[Bar96] BARAFF D.: Linear-time dynamics using Lagrange multipliers. In *Proc. of ACM SIGGRAPH* (1996).

[BBK05] BOTSCH M., BOMMES D., KOBBELT L.: Efficient linear system solvers for mesh processing. *IMA Mathematics of Surfaces XI, Lecture Notes in Computer Science 3604* (2005), 62–83.

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. In *Proc. of ACM SIGGRAPH* (2002).

[BNC96] BRO-NIELSEN M., COTIN S.: Real-time volumetric deformable models for surgery simulation using finite elements and condensation. *Computer Graphics Forum 15*, 3 (1996).

[CBC*05] CAPELL S., BURKHART M., CURLESS B., DUCHAMP T., POPOVIC Z.: Physically based rigging for deformable characters. *Proc. of Eurographics/ACM SIGGRAPH Symposium on Computer Animation* (2005).

[CGC*02] CAPELL S., GREEN S., CURLESS B., DUCHAMP T., POPOVIC Z.: Interactive skeleton-driven dynamic deformations. *Proc. of ACM SIGGRAPH* (2002).

[CHP89] CHADWICK J. E., HAUMANN D. R., PARENT R. E.: Layered construction for deformable animated characters. In *Proc. of ACM SIGGRAPH* (1989).

[CP03] CLINE M., PAI D.: Post-stabilization for rigid body simulation with contact and constraints. In *Proc. of IEEE International Conference Robotics and Automation* (2003), pp. 3744–3751.

[CW05] CIRAK F., WEST M.: Decomposition contact response (DCR) for explicit finite element dynamics. *International Journal for Numerical Methods in Engineering 64*, 8 (2005).

[DCKY02] DONG F., CLAPWORTHY G. J., KROKOS M. A., YAO J.: An anatomy-based approach to human muscle modeling and deformation. *IEEE Trans. on Visualization and Computer Graphics 8*, 2 (2002).

[Die06] DIEBEL J.: *Representing Attitude: Euler Angles, Quaternions, and Rotation Vectors*. Tech. rep., Stanford University, Palo Alto, CA, 2006.

[DSP06] DER K. G., SUMNER R. W., POPOVIC' J.: Inverse kinematics for reduced deformable models. In *ACM Transactions on Graphics* (July 2006), vol. 25, pp. 1174–1179.

[EDS05] ERLEBEN K., DOHLMANN H., SPORRING J.: The adaptive thin shell tetrahedral mesh. *Journal of WSCG* (2005), 17–24.

[EL00] EHMANN S. A., LIN M. C.: Accelerated proximity queries between convex polyhedra by multi-level voronoi marching. In *Proc. of International Conference on Intelligent Robots and Systems* (2000).

[Erl04] ERLEBEN K.: *Stable, Robust and Versatile Multibody Dynamics Animation*. PhD thesis, University of Copenhagen, 2004.

[Fea87] FEATHERSTONE R.: *Robot Dynamics Algorithms*. Kluwer, Boston, MA, 1987.

[Gas98] GASCUEL M.-P.: Layered deformable models with implicit surfaces. In *Proc. of Graphics Interface* (1998).

[GOM*06] GALOPPO N., OTADUY M. A., MECKLENBURG P., GROSS M., LIN M. C.: Fast simulation of deformable models in contact using dynamic deformation textures. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2006), pp. 73–82.

[GPS02] GOLDSTEIN H., POOLE C., SAFKO J.: *Classical Mechanics, 3rd Ed.* Addison Wesley, 2002.

[GTT89] GOURRET J.-P., THALMANN N. M., THALMANN D.: Simulation of object and human skin deformations in a grasping task. In *Proc. of ACM SIGGRAPH* (1989).

[GV96] GOLUB G. H., VAN LOAN C. F.: *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[GW05] GUO Z., WONG K. C.: Skinning with deformable chunks. *Proc. of Eurographics, Computer Graphics Forum 24*, 3 (2005), 373–382.

[HGB06] HALLQUIST J. O., GOURDREAU G., BENSON D. J.: TetGen. A quality tetrahedral mesh generator and three-dimensional Delaunay triangulator., 2006. http://tetgen.berlios.de.

[JP99] JAMES D. L., PAI D. K.: ArtDefo: accurate real time deformable objects. In *Proc. of ACM SIGGRAPH* (1999).

[JP02] JAMES D. L., PAI D. K.: Real time simulation of multizone elastokinematic models. In *IEEE International Conference on Robotics and Automation* (Washington, D.C., May 2002), pp. 927–932.

[JT05] JAMES D. L., TWIGG C. D.: Skinning mesh animations. In *Proc. of ACM SIGGRAPH* (2005).

[KJP02] KRY P., JAMES D. L., PAI D. K.: EigenSkin: Real time large deformation character skinning in hardware. In *Proc. of ACM SIGGRAPH Symposium on Computer Animation* (2002).

[KZ05] KAVAN L., ZARA J.: Spherical blend skinning: A real-time deformation of articulated models. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2005).

[LCF00] LEWIS J. P., CORDNER M., FONG N.: Pose space deformations: A unified approach to shape interpolation and skeleton-driven deformation. In *Proc. of ACM SIGGRAPH* (2000).

[MDM*02] MÜLLER M., DORSEY J., MCMILLAN L., JAGNOW R., CUTLER B.: Stable real-time deformations. *Proc. of ACM SIGGRAPH Symposium on Computer Animation* (2002).

[MG03] MOHR A., GLEICHER M.: Building efficient, accurate character skins from examples. In *Proc. of ACM SIGGRAPH* (2003).

[MT92] METAXAS D., TERZOPOULOS D.: Dynamic deformation of solid primitives with constraints. *Proc. of ACM SIGGRAPH* (1992).

[MTLT88] MAGNENAT-THALMANN N., LAPERRIERE R., THALMANN D.: Joint-dependent local deformations for hand animation and object grasping. In *Proceedings of Graphics Interface '88* (June 1988), pp. 26–33.

[OJSL04] OTADUY M. A., JAIN N., SUD A., LIN M. C.: Haptic display of interaction between textured models. In *Proc. of IEEE Visualization* (2004).

[PH06] PARK S. I., HODGINS J. K.: Capturing and animating skin deformation in human motion. In *Proc. of ACM SIGGRAPH* (2006).

[PPG04] PAULY M., PAI D. K., GUIBAS L. J.: Quasi-rigid objects in contact. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2004).

[SG06] SCHENK O., GÄRTNER K.: On fast factorization pivoting methods for symmetric indefinite systems. *Elec. Trans. Numer. Anal. 23* (2006), 158–179.

[Sha89] SHABANA A. A.: *Dynamics of Multibody Systems*. John Wiley and Sons, 1989.

[SK03] SONG P., KUMAR V.: Distributed compliant model for efficient dynamic simulation of systems with frictional contacts. In *Proc. of ASME Design Engineering Technical Conferences* (2003).

[TT93] TURNER R., THALMANN D.: The elastic surface layer model for animate character construction. In *Proc. of Computer Graphics International* (1993).

[TW88] TERZOPOULOS D., WITKIN A.: Physically based models with rigid and deformable components. *IEEE Computer Graphics and Applications 8*, 6 (1988).

[WTF06] WEINSTEIN R., TERAN J., FEDKIW R.: Dynamic simulation of articulated rigid bodies with contact and collision. *IEEE Trans. on Visualization and Computer Graphics 12*, 3 (2006).

[ZMCF05] ZORDAN V. B., MAJKOWSKA A., CHIU B., FAST M.: Dynamics response for motion capture animation. In *Proc. of ACM SIGGRAPH* (2005).

Appendix A: Kinematic Relationships

We can derive the world-frame velocity of a material point in terms of the velocity state vector by time differentiation of (2):

$$\begin{aligned}\dot{\mathbf{x}} &= \sum_{i=1}^{k} w_i(\dot{\mathbf{c}}_i - \mathbf{R}_i\mathbf{u}_i\omega_i + \mathbf{R}_i\mathbf{R}_{o,i}\mathbf{S}\dot{\mathbf{q}}_s)\\ &= \sum_{i=1}^{k} w_i(\dot{\mathbf{c}}_i + \mathbf{B}_i\dot{\theta}_i + \mathbf{R}_i\mathbf{R}_{o,i}\mathbf{S}\dot{\mathbf{q}}_s) \qquad (16)\end{aligned}$$

Each matrix $\mathbf{B}_i$ is the Jacobian of a vector $\mathbf{R}_i\mathbf{u}_i$ w.r.t. θ_i, with $\mathbf{u}_i$ a position in local bone space. The Jacobian can be proven to be equal to $-\mathbf{R}_i\tilde{\mathbf{u}}_i\mathbf{G}_i$ [Sha89], where $\tilde{\mathbf{u}}_i$ is the skew-symmetric cross-product matrix, and $\mathbf{G}$ relates local-frame angular velocities to time derivatives of quaternions through $\omega = \mathbf{G}\dot{\theta}$. We can rewrite (16) in compact matrix form as a linear function of the velocity state vector $\mathbf{v}$ (after application of $\dot{\mathbf{q}} = \mathbf{P}^{+}\mathbf{v}$ that encapsulates the adjoint relationship $\mathbf{G}$ from above, see [GOM*06] for details):

$$\dot{\mathbf{x}} = \mathbf{L}_W\mathbf{v} = \begin{bmatrix} \mathbf{W} & \mathbf{B}_W & \mathbf{R}_W\mathbf{S} \end{bmatrix}\mathbf{v}, \qquad (17)$$

$$\mathbf{B}_W = \begin{bmatrix} -w_1\mathbf{R}_1\tilde{\mathbf{u}}_1 & \dots & -w_k\mathbf{R}_k\tilde{\mathbf{u}}_k \end{bmatrix}, \quad \mathbf{R}_W = \sum_{i=1}^{k} w_i\mathbf{R}_i\mathbf{R}_{o,i},$$

and $\mathbf{W}$ is a diagonal weight matrix. $\mathbf{L}_W$ is position-dependent.

Appendix B: Joint Compliance for Hinge Joint

For a hinge joint aligned with axis of rotation $\mathbf{u}$, we model joint stiffness between bones i and j with an angular spring generating torques $\mathbf{T} = \pm k\theta\mathbf{u}$ proportional to the joint angle θ.

For implicit integration in (4), we also need the Jacobians $J = \frac{\partial \mathbf{T}}{\partial \mathbf{q}}$. We use quaternions $q = (s,x,y,z) = (q_s, \mathbf{q_u})$ to represent orientations and quaternion matrices [Die06] to represent quaternion multiplication: $\mathbf{q}_i \otimes \mathbf{q}_j = \mathbf{Q}(\mathbf{q}_i)\mathbf{q}_j = \bar{\mathbf{Q}}(\mathbf{q}_j)\mathbf{q}_i$:

$$\mathbf{Q}(\mathbf{q}) = \begin{pmatrix} s & -x & -y & -z \\ x & s & -z & y \\ y & z & s & -x \\ z & -y & x & s \end{pmatrix} = \begin{pmatrix} \mathbf{Q}_s \\ \mathbf{Q}_\mathbf{u} \end{pmatrix} \quad \bar{\mathbf{Q}}(\mathbf{q}) = \begin{pmatrix} s & -x & -y & -z \\ x & s & z & -y \\ y & -z & s & x \\ z & y & -x & s \end{pmatrix}$$

with $\mathbf{Q}_s \in \mathbb{R}^{1\times 4}$ and $\mathbf{Q}_\mathbf{u} \in \mathbb{R}^{3\times 4}$.

The difference orientation $\mathbf{q} = (q_s, \mathbf{q_u})$ between two quaternions $\mathbf{q}_i$ and $\mathbf{q}_j$ can be extracted with these quaternion matrices:

$$\begin{aligned} \mathbf{Q}_i = \bar{\mathbf{Q}}(\bar{\mathbf{q}}_i) \quad \mathbf{Q}_j \qquad &= \mathbf{Q}(\bar{\mathbf{q}}_j)\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \\ q_s = \mathbf{Q}_{j,s}\mathbf{q}_i = \mathbf{Q}_{i,s}\mathbf{q}_j \quad \mathbf{q_u} \qquad &= \mathbf{Q}_{j,\mathbf{u}}\mathbf{q}_i = \mathbf{Q}_{i,\mathbf{u}}\mathbf{q}_j \end{aligned}$$

Then, the Jacobians can be computed as follows:

$$\mathbf{J}_j^i = \frac{\partial \mathbf{T}_i}{\partial \mathbf{q_j}} = -\mathbf{J}_j^j = kz\big(w\mathbf{u}\mathbf{Q}_{i,s} + \theta\mathbf{Q}_{i,\mathbf{u}}\big)\frac{\mathbf{G}_j^T}{4}$$

$$\mathbf{J}_i^i = \frac{\partial \mathbf{T}_j}{\partial \mathbf{q_i}} = -\mathbf{J}_i^j = kz\big(w\mathbf{u}\mathbf{Q}_{j,s} + \theta\mathbf{Q}_{j,\mathbf{u}}\big)\frac{\mathbf{G}_i^T}{4}$$

$$\text{with} \quad z = cosec\big(\frac{\theta}{2}\big) \quad w = \frac{\theta}{tan(\theta/2)} - 2$$

For very small difference angles, we compute $\lim_{\theta\to 0}\mathbf{J}$:

$$\mathbf{J}_j^i = -\mathbf{J}_j^j = k\big(2\mathbf{Q}_{i,\mathbf{u}} - \frac{\theta}{2}\mathbf{u}^T\mathbf{Q}_{i,s}\big)\frac{\mathbf{G}_j^T}{4}$$

$$\mathbf{J}_i^i = -\mathbf{J}_i^j = k\big(2\mathbf{Q}_{j,\mathbf{u}} - \frac{\theta}{2}\mathbf{u}^T\mathbf{Q}_{j,s}\big)\frac{\mathbf{G}_i^T}{4}$$