Stable Integration of the Dynamic Cosserat Equations with Application to Hair Modeling

Gerrit Sobottka
Inst. of Computer Science II
Computer Graphics Group
University of Bonn
53117 Bonn, Germany
sobottka@cs.uni-bonn.de

Tomás Lay
Inst. of Computer Science II
Computer Graphics Group
University of Bonn
53117 Bonn, Germany
tomas@cs.uni-bonn.de

Andreas Weber
Inst. of Computer Science II
Computer Graphics Group
University of Bonn
53117 Bonn, Germany
weber@cs.uni-bonn.de

ABSTRACT

In this paper we propose a new method for stable numerical integration of the dynamic Cosserat equations for rods, which constitute a mechanical framework for the physically based modeling of slender structures like DNA strands, drill strings, marine cables or human hair. Our integration method is well-established in the field of structural dynamics and has the major advantage of unconditional stability as well as user controllable numerical damping. We demonstrate its advantages in the context of fiber-based modeling of human hair. To our knowledge this approach has not been used in the computer graphics community before.

Keywords

Cosserat theory, Cosserat rods, human hair, hair modeling, unconditionally stable, controllable numerical damping

1 INTRODUCTION

Recently the simulation of slender structures has gained increased attention in the computer graphics community. Typical examples are tubes, cords, catheters, or hair fibers. The special theory of Cosserat rods [Ant95] or so called director theories in general provide a mechanical framework for the description of the temporal evolution of such slender structures subject to external loads. Its major advantage is its completeness in theory as well as its ability to accurately describe local deformations like bend, twist, shear and extension. Our paper is motivated by the need for stable but fast numerical integration schemes for the dynamic Cosserat equations which arise in the context of the special theory of Cosserat rods [Ant95]. In particular, these equations constitute a system of coupled partial differential equations which are known to be stiff if certain material constraints are imposed.

While the static Cosserat equations have successfully been adopted in order to model the behavior of surgery cables [Pai02], human hair [SVW05][BAC+05][SW06] or flexible tubes in CAD applications [GS06] the dynamic Cosserat equations are relatively new to the computer graphics community and so effective solution methods are few and far between. In the special case of unshearable and inextensible rods they reduce to the well-known Kirchhoff equations. Recently, two models have been proposed [BAC+06][ST07] that approximate solutions to the dynamic Kirchhoff equations by restating the problem based on the Lagrange formalism. The resulting equations of motion together with the initial conditions are usually treated as an initial value problem. This is an obvious oversimplification because none of the approaches take the problem as it is: a system of non-linear coupled partial differential equations with boundary as well as initial conditions that together form a two point boundary value problem (BVP). BVPs are traditionally a domain of shooting or relaxation techniques.

In this context our specific contributions are as follows: We present a proper relaxation procedure for solving the dynamic Cosserat equations and demonstrate the efficiency of our approach by various examples. In particular, we introduce a relatively unknown implicit integration method coming from the field of structural dynamics. This so called *generalized α-method* is second-order accurate, unconditionally stable and allows for controllable numerical

damping. Furthermore, we show that finding solutions to the associated BVP need not necessarily be significantly slower than the numerical treatment of the simplified problem as an initial value problem .
We start with a brief discussion of the work related to our objective and proceed with an introduction of the theoretical framework of the special theory of Cosserat rods. In Sec. (4) a new stable integration method for stiff differential equation systems is introduced. Finally, we present some results to conclude with a discussion.

2 Related Work

Due to its importance we directly turn to the dynamic Cosserat equations and omit the discussion of solution methods to their static pendants. For the latter we refer the interested reader to the above cited literature.
In [BAC+06] an approximate solution method to the Kirchhoff equation in terms of Lagrange mechanics is presented wherein the system energies are described in terms of generalized coordinates, here the Darboux vector. The basic idea of this so called super-helix model is to approximate the underlying geometry of the curve by helices with piecewise constant curvature. This fact allows for relatively coarse spatial discretizations and enters the equations of motion by a simple selection function which makes the system discrete in space. Temporal discretization is then achieved by an implicit integration scheme. However, reduced coordinate formulations give rise to dense equation systems. This can become a bottleneck especially when refinements of the spatial discretization are necessary, e.g., if forces due to collision response or non-local electrostatic forces are acting on the structure. Moreover, collision detection on continuous models is normally not effective.
The Cosserat rod element approach [ST07] bases on Lagrange mechanics, too. Discrete energies are derived for piecewise linear segments of the rod with prescribed lengths. The major difference between both models—beside the linear and helical curve approximations—is that the latter enforces the constraints of unshearability and inextensibility by additional energy terms in the Lagrange equations while the former dislocates the problem to the explicit integration of the kinematic relations. The numerical integration of the kinematic relation as described in [ST07] is a little bit clumsy as the chosen type of integrator usually does not respect the basic properties of $\mathcal{SO}(3)$. Explicit integration of the kinematic relation using Rodrigues formula [Pai02] is always a good choice and we will also embark on this strategy.

3 Cosserat Theory

Let $\mathbf{r}(s,t) : [s_1, s_2] \times \mathbb{R} \mapsto \mathbb{R}^3$ be a smooth space curve of length L describing the centerline of the rod. Further, let $\{\mathbf{d}_i(s,t)\}$ be a set of orthogonal directors furnishing the space curve such that $\mathbf{d}_1$, $\mathbf{d}_2$ span the cross section plane and $\mathbf{d}_3 := \mathbf{d}_1 \times \mathbf{d}_2$ is orthogonal to it. The configuration at every time t is thus uniquely determined by the triple $\mathcal{C}(s,\cdot) = \{\mathbf{r}(s,\cdot), \mathbf{d}_1(s,\cdot), \mathbf{d}_2(s,\cdot)\}$. Please note that "orthogonal to the cross section plane" does not necessarily mean "tangent to the space curve" as the rod can undergo shear deformation.
From the existence of the local basis $\{\mathbf{d}_k\}$ it follows that there are vector valued functions $\boldsymbol{\omega}(s,t)$ and $\boldsymbol{\kappa}(s,t)$ that define the kinematic of the cross-section through

$$\partial_s \mathbf{d}_k = \boldsymbol{\kappa} \times \mathbf{d}_k, \tag{1}$$

$$\partial_t \mathbf{d}_k = \boldsymbol{\omega} \times \mathbf{d}_k. \tag{2}$$

Therefore, we call these equations *kinematic relations*. $\boldsymbol{\kappa}$ is the Darboux and $\boldsymbol{\omega}$ the twist vector. The kinematic of the center line is given by $\partial_s \mathbf{r} = \boldsymbol{\upsilon}$. All three quantities are decomposed with respect to the natural basis $\{\mathbf{d}_k\}$ as $\boldsymbol{\kappa} = \kappa_i \mathbf{d}_i$, $\boldsymbol{\omega} = \omega_i \mathbf{d}_i$, and $\boldsymbol{\upsilon} = \upsilon_i \mathbf{d}_i$. The Darboux vector $\boldsymbol{\kappa}$ and the vector $\boldsymbol{\upsilon}$ are the strain variables of the system and their components have physical meanings as they express the local deformation of the rod. The components of $\boldsymbol{\kappa} = (\kappa_1, \kappa_2, \kappa_3)$ measure bending (κ_1 and κ_2) and twist (κ_3) about the three directors while the components of $\boldsymbol{\upsilon} = (\upsilon_1, \upsilon_2, \upsilon_3)$ measure the shear (υ_1 and υ_2) and the extension (υ_3), respectively. In order to prevent the case of total shear in which the cross-section becomes parallel to the rod's centerline we require that $\upsilon_3 \equiv \mathbf{d}_3 \cdot \partial_s \mathbf{r} > 0$. This also implies the condition $|\partial_s \mathbf{r}| > 0$, so that the centerline cannot collapse to zero length.
In this context we require compatibility of the local basis $\{\mathbf{d}_k\}$,

$$\partial_t \partial_s \mathbf{d}_k(s,t) = \partial_s \partial_t \mathbf{d}_k(s,t), \tag{3}$$

which using Eq. (1) and Eq. (2) leads to the compatibility equation:

$$\partial_s \boldsymbol{\omega} = \partial_t \boldsymbol{\kappa} + \boldsymbol{\kappa} \times \boldsymbol{\omega}. \tag{4}$$

This means that the temporal evolution of the basis $\{\mathbf{d}_k\}$ must lead to the same orientation as its spatial evolution.

3.1 Balance Laws

The special theory of Cosserat rods [Ant95] used herein describes the evolution of a slender object subject to external loads. Conservation of linear and angu-

lar momentum leads us to the dynamic Cosserat equations which take the form

$$\partial_s \mathbf{n} + \mathbf{f} = \rho A \partial_{tt} \mathbf{r}, \tag{5}$$

$$\partial_s \mathbf{m} + \partial_s \mathbf{r} \times \mathbf{n} + \mathbf{l} = \rho \partial_t (\mathbf{I}\boldsymbol{\omega}), \tag{6}$$

where $\mathbf{n}$ and $\mathbf{m}$ are the contact force and contact couple at the cross sectional area, respectively. $\mathbf{f}$ and $\mathbf{l}$ are the external forces and moments acting on the rod. $\mathbf{I}$ the moment of inertia tensor, A is the area of the cross section and ρ the linear density which we assume to be constant.

3.2 Inextensibility - Infinite Stiffness

For our purposes we assume the rod to be hyperelastic, unshearable, and inextensible, thus, $\boldsymbol{\upsilon} = (0, 0, 1)$. This means that the tangent now coincides with the director $\mathbf{d}_3$ which is perpendicular to the cross section, i.e., $\partial_s \mathbf{r} = \mathbf{d}_3$. In our approach we implicitly enforce inextensibility by an appropriate constraint expressed through a differential equation. This is distinct from other approaches like e.g. [Pai02] where the inextensibility constraint is satisfied by explicit integration of the screw motion defined by the twist $(\boldsymbol{\kappa}, \boldsymbol{\upsilon})^T$. Due to continuity, we get the compatibility equation for the centerline $\mathbf{r}(s, t)$ as

$$\partial_t \partial_s \mathbf{r}(s, t) = \partial_s \partial_t \mathbf{r}(s, t). \tag{7}$$

Since $\mathbf{d}_3$ coincides with the tangent of the centerline it becomes locally independent of time (but not globally). Using the centerline velocity $\mathbf{u} = \partial_t \mathbf{r}$ our constraint takes the following form:

$$\partial_t \mathbf{d}_3 = \partial_s \mathbf{u}. \tag{8}$$

From a numerical point of view this equation is critical as it introduces severe stiffness into system. It acts like an infinite stiff spring and makes necessary the use of an implicit integration scheme. For this reason we will switch to a relatively unknown integration technique, the generalized α-method, cf. Sec. (4).

3.3 Material Laws

In order to relate the strain variables $\boldsymbol{\kappa}$ and $\boldsymbol{\upsilon}$ to the material internal forces $\mathbf{n}$ and torques $\mathbf{m}$ we need suitable constitutive relations of the form

$$\mathbf{m}(s, t) = \hat{\mathbf{m}}\left(\boldsymbol{\kappa}(s, t), \boldsymbol{\upsilon}(s, t), s\right), \tag{9}$$

$$\mathbf{n}(s, t) = \hat{\mathbf{n}}\left(\boldsymbol{\kappa}(s, t), \boldsymbol{\upsilon}(s, t), s\right). \tag{10}$$

Since the rod is unshearable and inextensible, i.e. $\boldsymbol{\upsilon} = (0, 0, 1)$, it is indicated to choose the material law for $\mathbf{m}$ as

$$\mathbf{m}(s, t) = \mathbf{K}(s)\, \Delta\boldsymbol{\kappa}(s, t), \tag{11}$$

which corresponds to Hook's law. Here, $\Delta\boldsymbol{\kappa} = (\boldsymbol{\kappa} - \hat{\boldsymbol{\kappa}})$ and $\hat{\boldsymbol{\kappa}}(s)$ is the initial deformation of the rod which does not depend on time. $\mathbf{K}(s) = \mathrm{diag}(EI_1(s), EI_2(s), G\mu(s))$ is a diagonal matrix describing the resistance of the material against bending and twist about the three axes. $I_{1,2}$ are the principal moments of inertia of the cross section A ($A \approx 1.5 \times 10^{-5}\,\mathrm{cm}^2$ for human hair fibers). Young's modulus E as well as the shear modulus G are empirical values that depend on the material (for Keratin $E = 3.89 \times 10^{10}$ and $G = 0.89 \times 10^{10}\,\mathrm{dynes/cm}^2$). If the cross-section is constant, $\mathbf{K}$'s dependency on s vanishes. Please note that there is no corresponding material law for $\mathbf{n}$, since shearing and dilatation of the rod are not allowed. In other words, there are no strains to which the contact couple $\mathbf{n}$ could be related. Thus, $\mathbf{n}$ can take any value that maintains the constraint $\boldsymbol{\upsilon} = (0, 0, 1)$.

3.4 Problem Reduction and Decoupling

The kinematic relation Eq. (1), the compatibility equation Eq. (4), the two balance laws Eq. (5) and Eq. (6), and the inextensibility constraint Eq. (8) together constitute a $(12+x)$-dimensional system of coupled PDEs, where x depends on the type of parameterization of the directors. Together with the boundary conditions (to be defined later) the task at hand is to solve a two point boundary value problem. We can simplify the problem by decoupling of the kinematic relation Eq. (1) from the system. This is done by decomposing all the equations with respect to the local basis $\{\mathbf{d}_k\}$. In other words, we treat the problem in local coordinates instead of global coordinates. This reduces the system to a 12-dimensional one because the kinematic relation Eq. (1) is integrated independently. Following Antman [Ant95] the worst thing that can happen is that the rotational invariance of the material laws gets lost. In the computer graphics context this was first used in [Pai02] where Pai introduces a rather approximate solution scheme to the static Kirchhoff equations by decoupling the kinematic relations from the equation system without explicitly pointing to this fact.
To reformulate the problem in local coordinates we use the fact that the derivate of any vector $\Diamond$ with respect to a fixed frame $\{\mathbf{e}_k\}$ is related to the local derivative by

$$\partial_t \Diamond^{\{\mathbf{e}_k\}} = \partial_t \Diamond^{\{\mathbf{d}_k\}} + \boldsymbol{\omega} \times \Diamond^{\{\mathbf{d}_k\}}, \tag{12}$$

$$\partial_s \Diamond^{\{\mathbf{e}_k\}} = \partial_s \Diamond^{\{\mathbf{d}_k\}} + \boldsymbol{\kappa} \times \Diamond^{\{\mathbf{d}_k\}}. \tag{13}$$

If we apply these rules to Eq. (4), Eq. (5), Eq. (6), and Eq. (8) we obtain the following final system of PDEs (please note that from now on all quantities are

referred to the local system):

$$\rho A \partial_t \mathbf{u} = \partial_s \mathbf{n} + \boldsymbol{\kappa} \times \mathbf{n} - \rho A \left(\boldsymbol{\omega} \times \mathbf{u}\right) + \mathbf{f}, \quad (14a)$$

$$\rho \mathbf{I} \partial_t \boldsymbol{\omega} = \partial_s \mathbf{m} + \boldsymbol{\kappa} \times \mathbf{m} + \mathbf{d}_3 \times \mathbf{n} - \rho \left(\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}\right) + \mathbf{l}, \quad (14b)$$

$$\partial_t \boldsymbol{\kappa} = \partial_s \boldsymbol{\omega} - \boldsymbol{\omega} \times \boldsymbol{\kappa} \quad (14c)$$

$$0 = \partial_s \mathbf{u} + \boldsymbol{\kappa} \times \mathbf{u} - \boldsymbol{\omega} \times \mathbf{d}_3. \quad (14d)$$

Here, we make use of the fact that—as mentioned above—due to the inextensibility constraint the local time derivative of the vector $\mathbf{d}_3$ must vanish. The contact torque $\mathbf{m}$ is expressed through $\boldsymbol{\kappa}$ using the material law Eq. (11). If we assume that the material parameters do not vary along s, the spatial derivative of $\mathbf{m}$ can be replaced by $\partial_s \mathbf{m} = \mathbf{K}(\partial_s \boldsymbol{\kappa} - \partial_s \hat{\boldsymbol{\kappa}})$. We omit the replacement of $\mathbf{m}$ and its derivative in the following equations for brevity.

The above equation system can be rewritten in the more concise form of the standard formulation for structural dynamics problems:

$$\hat{\mathbf{M}} \partial_t \mathbf{x}(s,t) + \hat{\mathbf{K}} \partial_s \mathbf{x}(s,t) + \boldsymbol{\Lambda}(s,t) = \mathbf{0}, \quad (15)$$

where $\mathbf{x}$ is the state vector of the rod, $\mathbf{x}(s,t) = \{\mathbf{u}(s,t), \boldsymbol{\omega}(s,t), \boldsymbol{\kappa}(s,t), \mathbf{n}(s,t)\}^T$, $\hat{\mathbf{M}} = \mathrm{diag}(\rho A,\ \rho A,\ \rho A,\ \rho I_1,\ \rho I_2,\ \rho I_3,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0)$ is the mass matrix and $-\hat{\mathbf{K}} = \mathrm{adiag}(\mathbf{1},\ \mathbf{1},\ \mathbf{K},\ \mathbf{1})$ is the stiffness matrix, with $\mathbf{K}$ as defined in Eq. (11). Note, that our mass matrix has not full rank since the time derivative of $\mathbf{n}$ does not exist. We collect all remaining terms in $\boldsymbol{\Lambda}(s,t)$. The above equation system is in the final form. We are now ready to integrate Eq. (14a) - Eq. (14d) in the form of Eq. (15). For this we adopt—as mentioned above—an implicit integration scheme from structural dynamics, the so called generalized α-method.

3.4.1 Boundary Conditions

To complete the system of equations we must specify six boundary conditions at both ends of the rod. Since our fibers are clamped at the scalp we have $BC_0 := \{\mathbf{r}(0,\cdot), \mathbf{d}_1(0,\cdot), \mathbf{d}_2(0,\cdot)\}$. Due to our coordinate free formulation we redefine this as $BC_0 := \{\mathbf{u}(0,\cdot), \boldsymbol{\omega}(0,\cdot)\}$ and express the "fixed"-condition by requiring that $\mathbf{u}(0,\cdot) = F_u(t)$ and $\boldsymbol{\omega}(0,\cdot) = F_\omega(t)$ with the functions $F_{(\cdot)}(t)$ taking non-zero values in case of moving supports only. At the opposite end we have $BC_L := \{\mathbf{m}(L,\cdot), \mathbf{n}(L,\cdot)\}$ which is expressed as $\{\boldsymbol{\kappa}(L,\cdot), \mathbf{f}(L,\cdot)\}$. Further, from the material law we know that $\boldsymbol{\kappa}(L,\cdot) = \mathbf{K}^{-1}\mathbf{m}(L,\cdot) + \hat{\boldsymbol{\kappa}}(L)$ and $\mathbf{m}(L,\cdot) = \mathbf{l}(L,\cdot)$ is equal to the external torque at $s = L$. Since there is no corresponding material law for $\mathbf{n}_L$ we assume that it is equal to the external force $\mathbf{f}(L,\cdot)$ acting at the end point of the rod. Other boundary conditions, e.g. for buckling can be integrated easily.

4 Generalized α-Method

In 1993 Chung and Hulbert introduced the generalized α-method for numerical integration of problems in structural dynamics [CH93]. In fact it is a Newmark-like implicit integrator with desirable features like second-order accuracy, unconditional stability and numerical dissipation of high-frequency noise. This typically occurs in stiff problems if the discretization of the domain is too coarse. In contrast to well-established implicit integration schemes in the computer graphics community, e.g., the implicit Euler method, the numerical damping is fully controllable by the user. While the idea of a controllable damping is not new in general the actual problem lies in achieving an optimal ratio between high-frequency and low-frequency dissipation, as the low-frequency portion is essential for the solution. The generalized α-method achieves this optimal ratio by a careful analysis of how the spectral radius of the amplification matrix behaves as certain system parameters grow to infinity. The spectral radius itself is a measure of the degree of numerical dissipation. The elimination of high-frequency noise can strongly improve the convergence behavior of non-linear problem solvers. Beside second-order accuracy and controllable damping the α-method has the nice property of being unconditionally stable. This means that we can take arbitrary large step sizes—in time as well as in space—without running the risk of becoming unstable. This does not only work in theory but also in practice, as we shall see later. The Kirchhoff rod model along with the α-method has been successfully applied by Sachin Goyal [GPL03] to model the structural mechanics of biomolecules. However, our solution method is different as we use a relaxation procedure instead of shooting.

4.1 Discretization of the System

We are now ready to discretize the system Eq. (14a) to Eq. (14d) in the form of Eq. (15). In the first step we derive the semi-discrete form of Eq. (15) with respect to time as

$$\hat{\mathbf{M}}^{1-\alpha_t} \partial_t \mathbf{x}^{1-\alpha_t} + \hat{\mathbf{K}}^{1-\beta_t} \partial_s \mathbf{x}^{1-\beta_t} + \boldsymbol{\Lambda}^{1-\beta_t} = \mathbf{0}. \quad (16)$$

The actual discretization of Eq. (15) by means of the α-method is straightforward as it follows a simple replacement rule: $\Diamond^{1-\varepsilon} := (1-\varepsilon)\,\Diamond^i + \varepsilon\,\Diamond^{i-1}$. This means that whenever the index $(1-\varepsilon)$ appears on a quantity of Eq. (15) we replace it with the averaging of this quantity over two subsequent time steps as given above, which leads to:

$$\begin{aligned} &\hat{\mathbf{M}}\left[(1-\alpha_t)\partial_t \mathbf{x}^i + \alpha_t \partial_t \mathbf{x}^{i-1}\right] \\ +&\hat{\mathbf{K}}\left[(1-\beta_t)\partial_s \mathbf{x}^i + \beta_t \partial_s \mathbf{x}^{i-1}\right] \\ &+\left[(1-\beta_t)\boldsymbol{\Lambda}^i + \beta_t \boldsymbol{\Lambda}^{i-1}\right] = \mathbf{0}. \end{aligned} \quad (17)$$

Here, we assume that the mass matrix as well as the stiffness matrix are constant. In the same spirit we apply the replacement rule to all quantities in Eq. (17) for spatial discretization to yield the final discrete form of the equation system

$$\begin{aligned}\hat{\mathbf{M}}\big\{&(1-\alpha_t)\left[(1-\alpha_s)\partial_t\mathbf{x}_j^i+\alpha_s\partial_t\mathbf{x}_{j-1}^i\right]\\&+\alpha_t\left[(1-\alpha_s)\partial_t\mathbf{x}_j^{i-1}+\alpha_s\partial_t\mathbf{x}_{j-1}^{i-1}\right]\big\}\\+\hat{\mathbf{K}}\big\{&(1-\beta_t)\left[(1-\beta_s)\partial_s\mathbf{x}_j^i+\beta_s\partial_s\mathbf{x}_{j-1}^i\right]\\&+\beta_t\left[(1-\beta_s)\partial_s\mathbf{x}_j^{i-1}+\beta_s\partial_s\mathbf{x}_{j-1}^{i-1}\right]\big\}\\&+\big\{(1-\beta_t)\left[(1-\beta_s)\boldsymbol{\Lambda}_j^i+\beta_s\boldsymbol{\Lambda}_{j-1}^i\right]\\&+\beta_t\left[(1-\beta_s)\boldsymbol{\Lambda}_j^{i-1}+\beta_s\boldsymbol{\Lambda}_{j-1}^{i-1}\right]\big\}=\mathbf{0}.\end{aligned}\tag{18}$$

In the above equation the index $\Diamond^i$ indicates the time step whereas the exponent $\Diamond_j$ indicates the node of the discretized curve. The left-hand side of Eq. (18) is a non-linear function $F_j(\mathbf{x}_j,\mathbf{x}_{j+1})=\mathbf{0}$ of the state vector $\mathbf{x}^i$ at the node j and the next node $j+1$. In Eq. (18) all quantities from the last time step $\Diamond^{i-1}$ are known while the $\mathbf{x}^i$ and $\boldsymbol{\Lambda}^i$ are to be computed. Note that $\boldsymbol{\Lambda}^i=\boldsymbol{\Lambda}^i(\mathbf{x}^i)$ is also a function of $\mathbf{x}^i$.
In order to solve the above equation system we must choose an appropriate approximation of the space and time derivatives of $\mathbf{x}$. If we use the trapezoidal rule we obtain the following recursive relationships for the time and space derivatives of the state vector $\mathbf{x}$:

$$\partial_t\mathbf{x}^i=\frac{\mathbf{x}^i-\mathbf{x}^{i-1}}{\gamma_t\Delta t}-\frac{1-\gamma_t}{\gamma_t}\partial_t\mathbf{x}^{i-1},\tag{19}$$

$$\partial_s\mathbf{x}_j=\frac{\mathbf{x}_j-\mathbf{x}_{j-1}}{\gamma_s\Delta s}-\frac{1-\gamma_s}{\gamma_s}\partial_s\mathbf{x}_{j-1},\tag{20}$$

where Δs and Δt are the spatial step and the time step, respectively and γ is some factor. For the derivative $\partial_s\hat{\boldsymbol{\kappa}}$ which appears in $\boldsymbol{\Lambda}$ we use the same approximation as for $\partial_s\mathbf{x}$, Eq. (20).

4.2 Stability and Numerical Damping

The stability of the generalized α-method depends on the choice of the coefficients α, β, γ. If the same discretization scheme is applied to the famous Dahlquist test equation, $\partial_t x+kx=0$, we can analyze the stability of the integrator through the spectral radius ρ which is the largest absolute eigenvalue $\rho(\mathbf{A})=\max|\lambda_{1,2}|$ of its amplification matrix $\mathbf{A}$, cf. [GG01]. The algorithm is unconditionally stable for linear problems if the spectral radius satisfies $\rho\leq 1$. For $\rho>1$ the solution grows from one time step to the next and the method becomes unstable. It was shown in [Gob00] that for the generalized α-method in order to remain stable it is required that $\alpha\leq 1/2,\beta\leq 1/2,\gamma\geq 1/2$ and for second order accuracy $\alpha-\beta+\gamma=1/2$.
The dissipation of high-frequency noise is controlled through the spectral radius as its magnitude is a measure for the degree of numerical dissipation. The spectral radius should be close to one for the low-frequency range and monotonically decreasing as $k\Delta t\rightarrow\infty$ approaches infinity. This requirement on the smoothness of the spectral radius restricts the possible range of eigenvalues and reduces the α-method to an one parameter method which is controlled by the eigenvalue λ^∞ for $k\Delta t$ at infinity (instead of $\lambda_{1,2}$). In the original work of Chung and Hulbert [CH93] it was shown that for a given value of $\lambda^\infty\in[-1,0]$ the coefficients of the α-method can be expressed as a function of this eigenvalue at infinity λ^∞ as: $\alpha=(3\lambda^\infty+1)/(2\lambda^\infty-2)$, $\beta=\lambda^\infty/(\lambda^\infty-1)$, and $\gamma=1/(1-\lambda^\infty)$, where $\alpha,\ \beta,\ \gamma\in[0,1]$. However, our differential equations are different from the Dahlquist test equation. As a consequence, these simple selection rules do not apply. In our case there is currently no known way for the computation of proper values for the integration parameters α, β, and γ. Since they are problem dependent finding the right parameters it is up to the user.

4.3 Solving the Equation System

With the above equations at hand we end up solving a non-linear equation system $\mathbf{F}:\mathbb{R}^{N\times 12}\mapsto\mathbb{R}^{(N-1)\times 12}$

$$\mathbf{F}(\mathbf{x})=\left\{\begin{array}{l}F_1\left(\mathbf{x}_1,\mathbf{x}_2\right)\\\vdots\\F_{N-1}\left(\mathbf{x}_{N-1},\mathbf{x}_N\right)\end{array}\right\}=\mathbf{0}\tag{21}$$

of size $(N-1)\times 12$ for the $N\times 12$ unknowns. Here, $\mathbf{x}^i=\{\mathbf{x}_1^i,..,\mathbf{x}_N^i\}$ is the global state vector of size $N\times 12$ at time step i containing the states $\mathbf{x}_j^i$ of all N nodes, where $j=1,...,N$. Together with the six boundary conditions at both ends the system is complete.
In order to solve the equation system we employ the classical Newton iteration, which linearizes the equation system in the area of the solution. Therefore, convergence is only achieved if the initial guess for the iteration is close to the solution. For the dynamic evolution this is usually not a problem as subsequent steps of the dynamic deformation are similar. Thus, we always take the configuration from the last time step as the initial guess for every new iteration.
The classical Newton approach uses the following iteration scheme:

$$\tilde{\mathbf{x}}^{k+1}=\tilde{\mathbf{x}}^k+\mathbf{J}^{-1}(\tilde{\mathbf{x}}^k)\mathbf{F}(\mathbf{x}^k),\tag{22}$$

where $\mathbf{J}$ is the Jacobian matrix of the system Eq. (21). The building blocks of $\mathbf{J}$ are the derivatives of Eq. (18) with respect to $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$.
In the same spirit as above we define a reduced version of the global state vector $\tilde{\mathbf{x}}=\{\mathbf{x}_1^i\backslash\mathrm{BC}_0,\mathbf{x}_2^i,...,$

$\mathbf{x}^i_{N-1}$,$\mathbf{x}^i_N \backslash \mathrm{BC}_L\}$ without the boundary conditions because we do not want to iterate over them. Thus, we assume that there exists a bijective function $\boldsymbol{\Xi} : \mathbb{R}^{N\times 12} \mapsto \mathbb{R}^{(N-1)\times 12}$ that transforms the global state vector to the reduced one and vice versa, $\tilde{\mathbf{x}} = \boldsymbol{\Xi}(\mathbf{x})$ and $\mathbf{x} = \boldsymbol{\Xi}^{-1}(\tilde{\mathbf{x}})$, by fading out/in the boundary conditions.

Rather than directly inverting the Jacobian matrix we solve the linear equation system $\mathbf{J}\delta\tilde{\mathbf{x}} = -\mathbf{F}(\mathbf{x}^k)$ and proceed with $\tilde{\mathbf{x}}^{k+1} = \tilde{\mathbf{x}}^k + \delta\tilde{\mathbf{x}}$. The Jacobian matrix is a tridiagonal block matrix with block size 12×12. Unfortunately, it is neither symmetric nor diagonal dominant so that the best choice for solving the above equation system is the Gauss elimination for sparse band matrices, cf. [PTVF02].

In order to improve stability of our Newton approach we never take the full step $\delta\tilde{\mathbf{x}}$ but pre multiply it with some relaxation parameter $\mu \in [0, 1]$. This under relaxation strongly improves the convergence behavior of our procedure. It is also possible to introduce an adaptive selection of the relaxation parameter like in [Gob00].

4.3.1 Derivation of the Jacobian Matrix

In what follows we aim to give a short derivation of the Jacobian matrix and its overall structure which may be a little bit unclear, especially how to deal with the boundary conditions. All we have to do is to take the derivatives $\partial_{\mathbf{x}_j} F_j$ and $\partial_{\mathbf{x}_{j+1}} F_j$ of all the functions in Eq. (21) with respect to the function variables $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$ or to their 12 components. Some care must be taken with the derivatives of the first and the last function, $\partial_{\mathbf{x}_1} F_1$ and $\partial_{\mathbf{x}_N} F_{N-1}$ as we have imposed six boundary conditions to each of the two state vectors. That is, the derivatives with respect to these variables vanish. In particular, we derive function F_1 with respect to the unknowns $\boldsymbol{\kappa}$ and $\mathbf{f}$ and function F_{N-1} with respect to $\mathbf{u}$ and $\boldsymbol{\omega}$. Hence, the first and the last building block of our global Jacobian has only size 12×6 whereas all other entries are of size 12×12. Thus, the global Jacobian has size $[(N-1)\times 12]^2$.

4.4 Integrating the Kinematic Relations

In Sec. (3.4) we decided to decouple the kinematic relations Eq. (1) from the system of governing PDEs in order to reduce the complexity of the problem. Since Eq. (1) is no longer part of the equation system it must be integrated separately. For this we use the well-known fact, that the integration yields a matrix exponential which can be solved by the application of Rodrigues formula. If the orientations are represented by rotation matrices then they are updated according to the following scheme [MLS94]:

$$\Delta\mathbf{R}^i_{j+1} = e^{\tilde{\boldsymbol{\kappa}}^i_j \Delta s}, \quad \mathbf{R}^i_{j+1} = \mathbf{R}^i_j \, \Delta\mathbf{R}^i_{j+1}, \qquad (23)$$

where $\tilde{\boldsymbol{\kappa}}$ is the skew symmetric matrix form of the Darboux vector. At the same time we can use the above rotational increment $\Delta\mathbf{R}_{j+1}$ to update the local forces $\mathbf{f}$ and torques $\mathbf{l}$ since the coordinate systems at the nodes change during the Newton iteration. For the sake of simplicity the spatial step size Δs is held constant.

On the other hand, for the new positions $\mathbf{r}^i_{j+1}$ we directly integrate the velocities $\mathbf{u}^i$ because equation Eq. (8) ensures that the new velocities satisfy the constraints on shear and extensibility. This is different to the explicit enforcement by considering spatial twists like in [Pai02].

4.5 Implementation Details

Direct implementation of the above equations in a programming language like C++ is an awkward and error prone task. To shorten the process of code generation we highly recommend the use of a computer algebra program like Maple or Mathematica to produce optimized source code, e.g. in C, for the evaluation of the left-hand side of Eq. (15) and the derivation of the Jacobian matrix. All we have to do is to type in Eq. (18) using the approximations Eq. (19) and Eq. (20) and invoke automatic differentiation of the latter with respect to $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$. The resulting C-code can be easily integrated into an existing framework as a subprocedure.

Now, let's assume that the code for the evaluation of the Jacobian matrix as well as the left hand side of Eq. (15) has been derived. Then our relaxation algorithm proceeds as follows: Given ρ^∞ (or α, β, γ), $\partial_s\hat{\boldsymbol{\kappa}}_0$ and $\partial_s\mathbf{x}^0_0$ we initialize the integration coefficients and the derivatives $\partial_s\hat{\boldsymbol{\kappa}}$, $\partial_s\mathbf{x}^0$ and $\partial_t\mathbf{x}^0$ as given above. This is done in the first time step only. Further, Eq. (15) is solved for $\partial_t\mathbf{x}^0$ since $\partial_t\mathbf{x}^{i-1}$ is an unknown in the first time step. Here, we encounter the problem that there exists no corresponding equation for the time derivative of the contact force $\partial_t\mathbf{n}^0_0$, so we choose it to be zero.

Now we are ready to iterate over the state vector $\mathbf{x}^k$ until a solution is found that zeros the function system Eq. (21). In each iteration we compute the Jacobian matrix and solve a sparse equation system in order to find the Newton update $\mathbf{x}^k$. With this vector we compute the new orientations according to Eq. (23) and transform the loads to the new local coordinate systems. Since our boundary conditions depend on the orientations as well they are updated in each iteration, too.

After convergence is achieved, we update the positions, which are normally not needed for the boundary conditions. If geometric boundary conditions are to be matched, e.g. a point-to-point constraint, the position update must be carried out within the iteration loop. The whole relaxation pseudo-code is given below.

Data: $i, \Delta t, \mathbf{x}^{i-1}, \mathbf{r}^{i-1}, \mathbf{R}^{i-1}, \mathbf{f}^i, \mathbf{l}^i$
Result: $\mathbf{x}^i, \mathbf{r}^i, \mathbf{R}^i$

if $i = 0$ then /* First time step */
  $\{\alpha, \beta, \gamma\} \leftarrow \mathsf{compCoefficients}(\rho^\infty)$;
  $\mathbf{x}^i \leftarrow \mathsf{updateBC}(\mathbf{x}^i)$;
  $\partial_s \hat{\boldsymbol{\kappa}} \leftarrow \mathsf{compDxDs}(\hat{\boldsymbol{\kappa}}, \partial_s \hat{\boldsymbol{\kappa}}_0, \Delta s, \gamma)$;
  $\partial_s \mathbf{x}^i \leftarrow \mathsf{compDxDs}(\mathbf{x}^i, \partial_s \mathbf{x}_0^i, \Delta s, \gamma)$;
  $\partial_t \mathbf{x}^i \leftarrow -\overline{\mathbf{M}}^{-1}(\overline{\mathbf{K}} \partial_s \mathbf{x}^i + \boldsymbol{\Lambda}^i)$;
end
$\mathbf{x}^{i-1} \leftarrow \mathbf{x}^i$; $\partial_s \mathbf{x}^{i-1} \leftarrow \partial_s \mathbf{x}^i$; $\partial_t \mathbf{x}^{i-1} \leftarrow \partial_t \mathbf{x}^i$;
$k \leftarrow 0$;
while $|\mathbf{F}(\mathbf{x}^i)| \geq \varepsilon$ do
  $\tilde{\mathbf{x}}^k \leftarrow \boldsymbol{\Xi}(\mathbf{x}^i)$;
  $\mathbf{J} \leftarrow \partial \mathbf{F}_m / \partial \tilde{\mathbf{x}}_n^k$;
  $\delta \tilde{\mathbf{x}} \leftarrow \mathsf{sparseSolve}\{\mathbf{J} \delta \tilde{\mathbf{x}} = -\mathbf{F}(\mathbf{x}^i)\}$;
  $\tilde{\mathbf{x}}^{k+1} \leftarrow \tilde{\mathbf{x}}^k + \delta \tilde{\mathbf{x}}$;
  $\mathbf{x}_M^i \leftarrow \boldsymbol{\Xi}^{-1}(\tilde{\mathbf{x}}^{k+1})$;
  $\mathbf{R}^{k+1} \leftarrow \mathsf{updateOrientations}(\mathbf{x}_M^i, \Delta s)$;
  $\{\mathbf{f}, \mathbf{l}\}^{k+1} \leftarrow \Delta \mathbf{R}^{k+1} \{\mathbf{f}, \mathbf{l}\}^k$;
  $\mathbf{x}^i \leftarrow \mathsf{updateBC}(\mathbf{x}_M^i)$;
  $\partial_t \mathbf{x}^i \leftarrow \mathsf{compDxDt}(\mathbf{x}^i, \mathbf{x}^{i-1}, \Delta t, \gamma)$;
  $\partial_s \mathbf{x}_0^i \leftarrow -\overline{\mathbf{K}}^{-1}(\overline{\mathbf{M}} \partial_t \mathbf{x}^i + \boldsymbol{\Lambda}^i)$;
  $\partial_s \mathbf{x}^i \leftarrow \mathsf{compDxDs}(\mathbf{x}^i, \partial_s \mathbf{x}_0^i, \Delta s, \gamma)$;
  $k \leftarrow k + 1$;
end
$\mathbf{r}^i \leftarrow \mathsf{updatePositions}(\mathbf{x}^i, \Delta s)$;
$\{\mathbf{f}, \mathbf{l}\}^{i-1} \leftarrow \{\mathbf{f}, \mathbf{l}\}^i$

Algorithm 1: Relaxation procedure for computing solutions to the equations of motion for the dynamic Cosserat rod. $\mathbf{F}(\mathbf{x})$ is the left-hand-side of the discretized system Eq. (21).

5 Results

For our numerical experiments a test environment was written in C++. As stated above, the numeric code was generated using Maple. We consider different test cases each of which is consisting of 100 segments: **1.)** A sinus-like shaped rod which is released under gravity from a horizontal position (no damping). **2.)** A highly damped helical rod (30 cm) subject to a time-varying end point load. The damping is obtained by setting $\alpha = 0.4$, $\beta = 0$, and $\gamma = 1.0$. **3.)** A straight rod (45 cm) subject to a time-varying torque. **4.)** A helical rod with low damping that is excited by a force parallel to the axis of the helix and released after 0.1 secs. This example shows the typical oscillating behavior of a steel-like coil spring. Furthermore, we varied the number of segments from six up to 100 and in a second experiment the time step Δt from 10^{-5} up to 100 secs. Interestingly, our algorithm remains stable, regardless of how many segments we use for discretization. The same holds for the time step. This result clearly shows that the prime property of unconditional stability of the generalized α-method applies not only in theory but also in practice. **5.)** The buckling behavior of a slender structure is one of the canonical examples in rod mechanics. Therefore, we demonstrate that our model is also capable of capturing this important effect. For this the end-points of a straight rod are being moved towards each other while it experiences an axial torque. The movement of the rod accelerates as the solution approaches its bifurcation point. **6.)** In the sixth example we present a hair tress (25 cm) consisting of a deformable guide (Keratin) and 180 interpolated fibers. The guide is extended by applying a constant velocity to its end and then released after a total extension of 30 % of its bounding box length. The effects are very similar to that of a force driven simulation, e.g. a hair tress under gravity. But in the former case the end point is constrained to a line parallel to the direction of gravity. For this the boundary conditions at $s = L$ have to be modified according to $BC_L := \{\mathbf{v}, \boldsymbol{\omega}\} = \{\lambda d_{13}, \lambda d_{23}, \lambda d_{33}, 0, 0, 0\}$ where we choose the extension speed $\lambda = -2.0$. This shows how easily the boundary conditions can be adopted to a given problem. **7.)** Hair tress under gravity and its kinetic energy for different values of α and γ ($\beta = 0$). The results are depicted in Fig. (1).
Our relaxation procedure converges within 14 iterations on the average. The time until convergence for 100 segments is approximately 135 ms on a laptop with Intel Pentium M (Centrino), 1.86 GHz and 1GB RAM and 'standard' boundary conditions as introduced above. The time step was $\Delta t = 1/30$ in all examples and $\alpha = 0.3$, $\beta = 0$, and $\gamma = 0.7$.

5.1 Conclusion

We have presented a novel approach for the simulation of slender structures based on the special theory of Cosserat rods that is fast and stable. In contrast to existing approaches, that are based on simple Lagrange mechanics we directly solve the system of governing PDEs using the generalized α-method. Our method demonstrates that there is an efficient way to deal with the Cosserat equations, not necessarily through model simplifications, and proves also the fact that integration methods from structural engineering have applications in the computer graphics field as well. In this spirit our model must be seen as the first approach that tackles the problem at its root. Our approach is well-suited especially for the simulation of human hair. In combination with proper contact models it should be possible to analyze all the physical effects observed in hair fiber interaction. Moreover, we think that our approach can be successfully adopted to other problem areas like cloth modeling as well.

References

[Ant95] Stuart S. Antman. *Nonlinear Problems of Elasticity*, volume 107 of *Appl. Math. Sci.* Springer-Verlag, Berlin and New York, 1995.

[BAC+05] Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. Predicting natural hair shapes by solving the statics of flexible rods. In J. Dingliana and F. Ganovelli, editors, *Eurographics (short papers)*, August 2005.

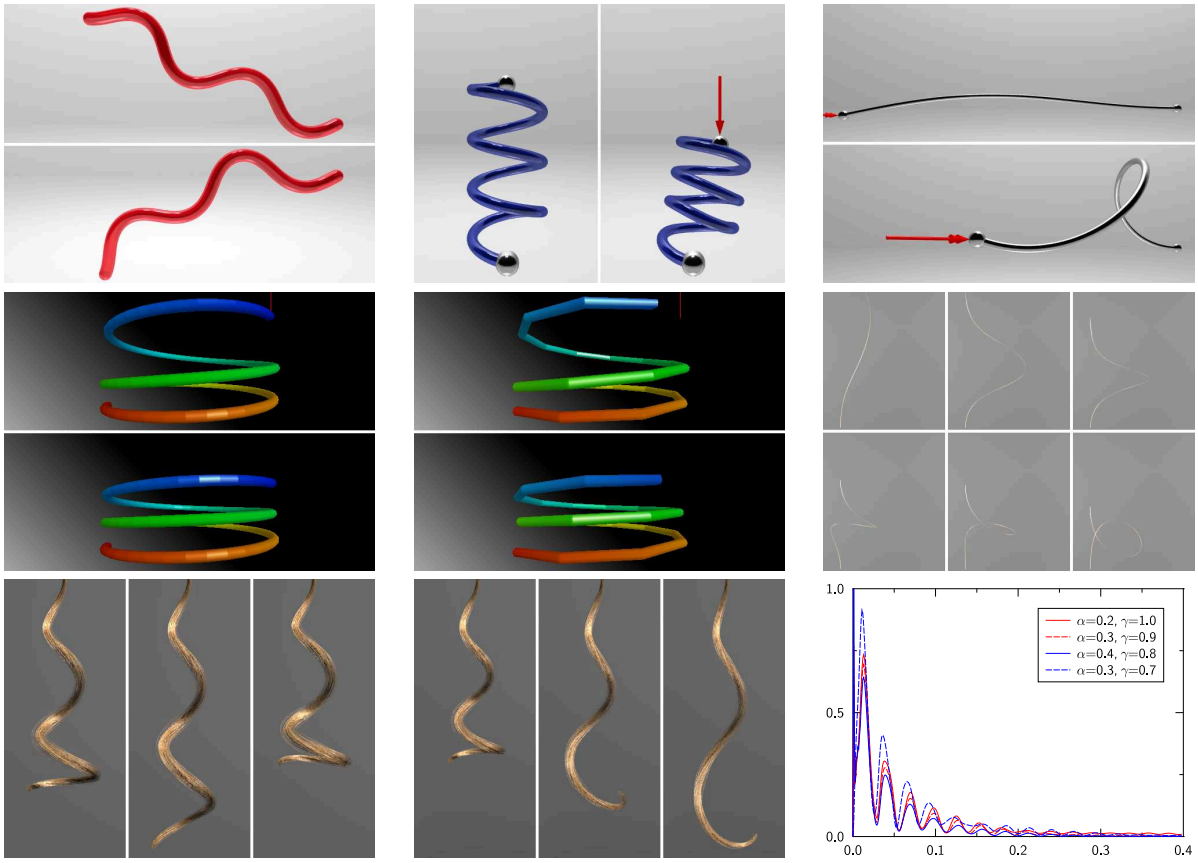

Figure 1: Various examples computed with our relaxation technique (100 segments each): **Top row**: 1.) Non-straight rod subject to gravity. 2.) Helical rod with high damping subject to end point load. 3.) Straight rod subject to end point torque. **Center row**: 4.) Coil spring with low damping which is excited by an vertical end point load and released after 0.1 secs (100 and 25 segments). *Right*: 5.) The canonical example: buckling. **Bottom row**: 6.) Snapshots from the velocity driven extension of a hair tress (25 cm). The tress snaps back after a total elongation of 30 % of the bounding box length. 7.) Hair tress under gravity and its kinetic energy ($\beta = 0$).

[BAC+06] Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. Super-helices for predicting the dynamics of natural hair. *ACM Trans. Graph.*, 25(3):1180–1187, 2006.

[CH93] J. Chung and G.M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation - the generalized-alpha method. *J. Appl. Mech.*, 60(6):371–375, 1993.

[GG01] J.I. Gobat and M.A. Grosenbaugh. Application of the generalized-alpha method to the time integration of the cable dynamics equations. *Computational Methods in Applied Mechanics and Engineering*, 190(37):4817–4829, June 2001.

[Gob00] Jason I. Gobat. *The Dynamics of Geometrically Compliant Mooring Systems*. Phd-thesis in oceanographic engineering, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution, 2000.

[GPL03] S. Goyal, N. C. Perkins, and C. L. Lee. Torsional buckling and writhing dynamics of elastic cables and DNA. In *Proceedings of ASME 2003 Design Engineering and Technical Conference, Chicago, USA, September 26*, 2003.

[GS06] Mireille Grégoire and Elmar Schömer. Interactive simulation of one-dimensional flexible parts. In *Proceedings of the 2006 ACM symposium on Solid and physical modeling*, pages 95–103, 2006.

[MLS94] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[Pai02] Dinesh K. Pai. STRANDS: Interactive simulation of thin solids using Cosserat models. In *Computer Graphics Forum*, volume 21(3), pages 347–352, 2002.

[PTVF02] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press; 2nd Edition, February, 2002.

[ST07] Jonas Spillmann and Matthias Teschner. CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 63–72, 2007.

[SVW05] Gerrit Sobottka, Ebadollah Varnik, and Andreas Weber. Collision detection in densely packed fiber assemblies with application to hair modeling. In *Proceedings of the Conference on Imaging Science, Systems and Technology: Computer Graphics, Las Vegas, 2005*, pages 244–250, 2005.

[SW06] Gerrit Sobottka and Andreas Weber. Computing static electricity on human hair. In *Proceedings of Theory and Practice of Computer Graphics 2006*, pages 21–29, 2006.