Hair Simulation Model for Real-Time Environments

P. Knoch¹ and U. Bonanni² and N. Magnenat-Thalmann²

¹Charles University in Prague, Faculty of Mathematics and Physics, Czech Republic

²MIRALab, University of Geneva, Switzerland

Abstract

We present a dynamic hair animation method designed for use in real-time virtual environments. Basing on recent research results in elastic rod simulation, we introduce an enhanced physically plausible technique which utilizes specific properties of hair strands to achieve better performance and more stable simulation. Stability is also enhanced by decoupling major sources of dynamic equation stiffness into a separate post-integration step. We also use the same decoupling principle to treat hair-head collisions in a very efficient and robust way. Thanks to this, we can afford to use a fast explicit integration scheme, which, combined with a short time step, allows our method to provide smooth results even in frequency-sensitive areas such as haptics-based hair modelling.

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

---

1. Introduction

Realistic animation and rendering of hair is a crucial part of presenting virtual humans. Without naturally-behaving hair, realism of the presented scenes is severely hampered, as the head of a virtual human forms a natural focal point for the observers. Unfortunately, realistically animating hair is no easy task, as it exhibits several specific physical characteristics. Hair strands have a naturally anisotropic character; the length of a typical strand is several orders of magnitude larger than its diameter. Hair is also practically unstretchable and unshearable. At the same time it bends and twists easily, but resumes its rest shape when external strain is removed. These properties, combined with the fact that a typical human has over 100,000 individual hair strands, make accurate and fast physical simulation very difficult.

Goals and contributions We concentrate on animating hair in real-time scenarios (like haptics), giving up strict realism in exchange for speed, but maintaining physical basis and plausibility. To this effect, we base our approach on the model introduced in [BWR*08], originally designed for larger, very flexible objects such as ropes. All potential sources of equation stiffness are moved from the integration itself into a post-processing step. This, combined with a short time step of 1ms, allows the use of a fast explicit integration scheme which is computable in real time. That way,

we also gain the option to render simulation results at almost arbitrary frequency — by using a 1kHz simulation speed, we decouple simulation from rendering and make sure even a fast rendering system gets smooth data.

Our main contribution lies in the development of a new method to handle twisting. Utilizing specific properties of hair strands, our method is both faster and more robust. Simulating correct hair behaviour requires stiffnesses 2–3 orders of magnitude larger than most scenarios depicted in [BWR*08]; our twist method is fully capable of dealing with such stiffnesses neither (a) being slowed down nor (b) requiring further shortening of the simulation time step. In addition, our method involves no matrix iterations, thus having a smaller memory footprint, and is easily parallelizable.

Another important result is extending the method's approach to constraint enforcement to solve hair-head collisions in a robust way with just minimal computational overhead.

Organization The rest of this paper is organized as follows: Section 2 presents an overview of current methods used for dynamic hair animation. Section 3 gives a brief outline of our simulation. Section 4 describes the physical model on which we base our simulation, with its discretization and modifications given in Sections 5 and 6. Details of the simulation itself are given in Section 7, while Section 8

describes our constraint- and collision-handling system. Finally, Section 9 presents our results and concludes the paper.

2. Related work

Methods for animating hair fall into two general categories: strand based and volume based. The idea to represent hair as a volume was first introduced in [HMT01]. Hair is treated as a volume of “hair matter” and simulated using continuum mechanics implemented by smoothed particles. Individual hair strands are still retained, expressed as a chain of rigid links, with joints glued to continuum particles.

Further volumetric methods abandon the notion of individual strands even more. In [VMT04], hair volume is modelled as a free-form lattice with nodes acting as simulated particles. Strands are attached to the lattice as viscoelastic springs. The most radical approach was adopted in [BCN03], where hair is simulated as smoothed particles loosely connected by springs, with no notion of strands. Appearance of hair is obtained using texture splatting.

Volumetric methods generally achieve faster simulation, but tend to produce uniform deformations and cannot capture complex hair behaviour. The other approach is to model hair explicitly, either as individual strands or as wisps. One framework used for modelling strand dynamics are mass-spring systems. Different models of several springs are used to capture complex hair behaviour such as twisting [SLF08] or effects of styling products [WGL04]. Another paradigm employed are rigid multi-body chains [CCK05, Had06, WL03]. A drawback of these methods is long computation time, which prevents them from use in real-time applications (this can be helped by localizing simulation [WGL06] or using LOD hierarchies [WL03]).

A different dynamic model is the Cosserat theory of elastic rods, first introduced into computer graphics in [Pai02]. An application of this theory to hair simulation was presented in [BAC*06], using Kirchhoff equations discretized as *super-helices*, a piecewise helical structure. Using a helix as a simulation primitive greatly reduces the number of variables, but the resulting system is non-linear and expensive to compute. Furthermore, as strands are reconstructed implicitly, modelling hair-object interactions is difficult.

Kirchhoff rods have also recently been studied in areas not related to hair [BWR*08, ST07]. Both use explicit centreline representation of nodes and segments. The method of [ST07] represents material torsion using quaternions, while in [BWR*08], scalar rotation from a reference frame is used and twist is treated quasi-statically. For low-stiffness materials, these methods reach interactive simulation rates.

3. Algorithm outline

Our system simulates hair on a per-strand basis. The entire hair volume is viewed as a collection of individual *leader*

strands, subject to physical simulation, and a greater number of *follower* strands, the state of which is just interpolated from leaders (see Figure 1(b)). This keeps the number of simulated strands at a manageable level, while still allowing non-uniform behaviour in the hair volume.

An outline of our method is presented as Algorithm 1. Notice the decoupling of all possible sources of equation stiffness to post-integration steps (steps 5–10).

Algorithm 1: Hair simulation outline

```

1 precompute rest-state values;
2 while simulation running do
3   compute forces;
4   integrate equations of motion;
5   detect hair-head collisions;
6   while constraints or collisions unsolved do
7     perform one constraint enforcement step;
8   if positions changed then update velocities;
9   update Bishop frame;
10  compute twist;
11 end;
```

---

4. Physics of Kirchhoff rods

4.1. Representation

We treat hair as individual strands, modelled as *elastic rods*. A rod is a deformable body whose one dimension (length) is significantly larger than the other two (cross section). The configuration of a rod is entirely described by the position of its *centreline*, its *cross section shape* and its *material torsion*. Thus, a rod $\Gamma(s) = \{\mathbf{x}(s), \mathbf{m}_1(s), \mathbf{m}_2(s)\}$, where $\mathbf{x}(s)$ is centreline position and $\mathbf{m}_{1,2}(s)$ are axes of the cross section. s runs from 0 to the rod’s length L .

For us each rod represents one hair strand, which is *inextensible* and *unshearable*. Unshearability is embedded directly into our model: we introduce a unit vector $\mathbf{t}(s)$ tangent to the centreline, $\mathbf{t}(s) \parallel \mathbf{x}'(s)$, and require $\{\mathbf{t}, \mathbf{m}_1, \mathbf{m}_2\}$ to form an orthonormal frame, the *material frame*. (The prime accent denotes differentiation along the centreline: $a' = \frac{\partial a}{\partial s}$. The dot accent stands for differentiation by time: $\dot{a} = \frac{da}{dt}$).

Unfortunately, inextensibility cannot be treated the same way, as this would result in stiff equations. To avoid this, we neither model nor enforce inextensibility in the physical simulation itself, but delegate its enforcement to a post-integration step (see Section 8).

4.2. Elastic energy

The elastic energy $E(\Gamma)$ of the rod, as given by the Kirchhoff theory of elastic rods, is composed of a bending and twisting component (because of our assumption of inextensibility, there’s no stretching component). The energy depends on

Figure 1: Curly hair in a rest state which exhibits natural twist (a); same configuration with leaders highlighted (b). The left-hand followers copy their leader, while the right-hand ones are interpolated from the two leaders, based on the leaders' area of influence.

strain: the rate of change of the material frame, expressed in the frame itself:

$$\omega_1 = \mathbf{t}' \cdot \mathbf{m}_1, \quad \omega_2 = \mathbf{t}' \cdot \mathbf{m}_2, \quad m = \mathbf{m}'_1 \cdot \mathbf{m}_2$$

If we denote the centreline curvature $\mathbf{\kappa} = \mathbf{t}'$, we can see that $\omega_{1,2}$ correspond to the rod *bending* over the cross section axes $\mathbf{m}_{1,2}$, while m measures *twist* of the material frame.

We denote the rod's bending $\boldsymbol{\omega} = (\omega_1, \omega_2)^T$. The bending energy of the rod depends on its cross-sectional bending properties and its deviation from natural bending:

$$E_{\text{bend}} = \frac{1}{2} \int_0^L (\boldsymbol{\omega}(s) - \hat{\boldsymbol{\omega}}(s))^T \hat{B}(s) (\boldsymbol{\omega}(s) - \hat{\boldsymbol{\omega}}(s)) ds$$

$\hat{B}$ is a 2×2 quadratic form (a symmetric positive-definite matrix) which describes the rod's bending behaviour. $\hat{\boldsymbol{\omega}}$ is the bending of the rod in its natural state. (We use the hat accent $\hat{a}$ to denote a quantity that is precomputed from the rod rest state before simulation start). Analogously, the twisting energy is defined as:

$$E_{\text{twist}} = \frac{1}{2} \int_0^L \beta(s) (m(s) - \hat{m}(s))^2 ds$$

where β is twisting stiffness. Unlike the original model, we also consider the rod's rest-state twist, $\hat{m}$, to allow for naturally twisted rods. This allows us to model styled hair, which usually involves a twisted rest state (curls). See Figure 1(a) for an example of such a rest state. The above formula relates the twisting energy to material frame vectors. Following the ideas presented in [BWR*08], we borrow some concepts from differential geometry to express it using fewer variables.

Bishop frame and parallel transport As twist is a scalar, we would like to be able to express it using a scalar variable; i.e. express the material frame as a rotation of a twist-free reference frame. An adapted frame $\{\mathbf{t}(s), \mathbf{u}(s), \mathbf{v}(s)\}$ with no twist is called a *Bishop frame*. It is uniquely defined by fixing $\mathbf{u}(s_0)$ and $\mathbf{v}(s_0)$ for any one s_0 . In our model, the most convenient point is at the root, $s_0 = 0$. The evolution of the Bishop frame along the centreline can be expressed using its *Darboux vector* $\boldsymbol{\Omega}(s)$:

$$\mathbf{t}' = \boldsymbol{\Omega} \times \mathbf{t}, \quad \mathbf{u}' = \boldsymbol{\Omega} \times \mathbf{u}, \quad \mathbf{v}' = \boldsymbol{\Omega} \times \mathbf{v}$$

As the frame is twist free, $\boldsymbol{\Omega}$ has no tangential component, and thus $\boldsymbol{\Omega} = \mathbf{t} \times \mathbf{\kappa} = \mathbf{\kappa} \mathbf{b}$, which is the *curvature binormal*.

The Darboux vector of the Bishop frame is used to define *parallel transport*. Parallel transport of a vector $\mathbf{a}$ from one point of the centreline to another is achieved by integrating the equation $\mathbf{a}' = \mathbf{\kappa} \mathbf{b} \times \mathbf{a}$. Infinitesimally, this corresponds to a rotation about the binormal. Parallel transport carries out a twist-free evolution of vector $\mathbf{a}$. By definition, the axes of the Bishop frame evolve using parallel transport.

Twist representation Let us define a scalar function $\theta(s)$ which measures the angle (around the tangent) between the material frame and the Bishop frame:

$$\mathbf{m}_1 = \cos(\theta) \mathbf{u} + \sin(\theta) \mathbf{v}, \quad \mathbf{m}_2 = -\sin(\theta) \mathbf{u} + \cos(\theta) \mathbf{v}$$

This allows us to express twist as $m(s) = \theta'(s)$. Thus, we have expressed the rod's elastic energy using 4 dimensions: a 3-dimensional centreline position $\mathbf{x}(s)$ and a scalar angle between the Bishop and material frame $\theta(s)$.

5. Model discretization

5.1. Hair specifics

To obtain a discrete representation of an elastic rod, we follow the ideas presented in [BWR*08]. However, as our model is intended for hair, we also consider several aspects specific to hair, which allows us to simplify the model.

One of the most important considerations is that hair strands generally have an elliptical cross section. Because of this, they tend to bend over their cross section major axis only [Swi95]. This idea has been used to directly reduce the degrees of freedom of an implicit hair model in [BK08]. In our explicit model, such a direct approach is not possible. Nevertheless, we can use this idea to guide our twist computation, as major axis directions (and thus twist) are effectively determined by centreline bending. Section 6 deals with this approach in detail.

5.2. Discrete representation

Given a rod $\Gamma(s) = \{\mathbf{x}(s), \mathbf{m}_1(s), \mathbf{m}_2(s)\}$, we discretize the centreline into $n + 2$ nodes $\mathbf{x}_0, \dots, \mathbf{x}_{n+1}$ connected by $n + 1$ segments $\mathbf{e}^0, \dots, \mathbf{e}^n$. Throughout this paper, we use lower

Figure 2: Strand discretized into 30 segments of 1cm length. Nodes and segment material frames shown (a), material frames in detail (b)

indices for quantities assigned to nodes and upper indices for those assigned to segments.

We assign a material frame $\{\mathbf{t}^j, \mathbf{m}_1^j, \mathbf{m}_2^j\}$ to each segment j (see Figure 2). Note that such an assignment is unique, whereas assigning a tangent to a polygonal curve's node is generally ambiguous. We keep the requirement of the frame being adapted to the centreline, meaning $\mathbf{t}^j = \frac{\mathbf{e}^j}{|\mathbf{e}^j|}$.

Integrated quantities As shown in [BWR*08], a distinction must be made between quantities defined pointwise and those representing a value *integrated* over a domain. When an integrated quantity is associated with a node, its domain are the nearest halves of segments adjoining to the node. For node $\mathbf{x}_i$, the domain has length $l_i/2$, where $l_i = |\hat{\mathbf{e}}^{i-1}| + |\hat{\mathbf{e}}^i|$.

5.3. Discrete elastic energy

Bending energy Discrete curvature binormal (an integrated quantity) and bending can be expressed as:

$$(\kappa\mathbf{b})_i = 2 \frac{\mathbf{e}^{i-1} \times \mathbf{e}^i}{|\hat{\mathbf{e}}^{i-1}| |\hat{\mathbf{e}}^i| + \mathbf{e}^{i-1} \cdot \mathbf{e}^i}$$

$$\boldsymbol{\omega}_i^j = \left((\kappa\mathbf{b})_i \cdot \mathbf{m}_2^j, -(\kappa\mathbf{b})_i \cdot \mathbf{m}_1^j \right)^T \text{ for } j \in \{i-1, i\}$$

Note that there are two bending values (two 2-vectors) for each node, one expressed in each adjoining segment's material frame. These allow us to derive the following formula for bending energy:

$$E_{\text{bend}}(\Gamma) = \sum_{i=1}^n \frac{1}{2\hat{l}_i} \sum_{j=i-1}^i \left(\boldsymbol{\omega}_i^j - \hat{\boldsymbol{\omega}}_i^j \right)^T \hat{B}^j \left(\boldsymbol{\omega}_i^j - \hat{\boldsymbol{\omega}}_i^j \right)$$

As discussed in Section 5.1, hair strands only bend over

their cross-sectional major axis $\mathbf{m}_1$. We utilize this limitation by combining an appropriate twisting method (Section 6) with a specifically crafted bending stiffness matrix B^j :

$$B^j = \begin{pmatrix} \mu\alpha^j & 0 \\ 0 & \alpha^j \end{pmatrix}$$

Here, α^j is bending stiffness of the strand on segment j , while μ is an arbitrary constant used to penalize bending over the minor axis. Its use is discussed in Section 6.2.

Bishop frame To define the Bishop frame, we need discrete parallel transport. We define it as a rotation P_i around the curvature binormal, or identity if $(\kappa\mathbf{b})_i = \mathbf{0}$. We then fix the value of vector $\mathbf{u}^0$ (at the root) and parallel transport it along the rod, thus obtaining $\mathbf{u}^i = P_i(\mathbf{u}^{i-1})$; then set $\mathbf{v}^i = \mathbf{t}^i \times \mathbf{u}^i$.

Twisting energy With the Bishop frame defined, we can express the material frame of segment j using a rotation θ^j in a straightforward analogy of the continuous case:

$$\mathbf{m}_1^j = \cos \theta^j \mathbf{u}^j + \sin \theta^j \mathbf{v}^j, \quad \mathbf{m}_2^j = -\sin \theta^j \mathbf{u}^j + \cos \theta^j \mathbf{v}^j$$

The twisting energy expression follows naturally:

$$E_{\text{twist}}(\Gamma) = \sum_{i=1}^n \beta \frac{(m_i - \hat{m}_i)^2}{\hat{l}_i}, \text{ where } m_i = \theta^i - \theta^{i-1}$$

6. Material frame update

6.1. Quasistatic twist treatment

One of the important principles of the model described in [BWR*08] is *quasistatic* treatment of twist. As twist waves propagate much faster than bending ones, they can safely be treated by a static update inbetween simulation frames and thus *excluded from the simulation itself*.

The strand can be *clamped* at one or more segments $C = \{c(1), \dots, c(p)\}$; the material frame position is prescribed at those segments. Normally, a hair strand is only clamped at the root, $C = \{0\}$; it is possible to clamp other segments as well, for example during user interaction (simulating a strand being held and thus unable to twist freely).

Bishop frame update When a simulation step finishes, before twisting can be computed, the Bishop frame must be updated. In general, after a simulation step t_s , it can happen that $\mathbf{u}^0 \neq \mathbf{t}^0$. To re-align the Bishop frame, we compute a rotation ${}_tP$ which parallel transports vectors *in time* so that ${}_tP(\mathbf{t}^0(t_{s-1})) = \mathbf{t}^0(t_s)$. ${}_tP$ is then used to parallel transport $\mathbf{u}^0(t_{s-1})$ to $\mathbf{u}^0(t_s)$. The Bishop frame of the rest of the segments is then updated using normal parallel transport P_i .

Material frame update With the Bishop frame updated, we can compute new twist, thereby updating the material frame. The strand twists to minimize its internal energy, unclamped segments therefore rotate to minimize $E(\Gamma)$. The

methods presented in [BWR*08] uses Newton minimization to find this new twist. In our case, we are dealing with hair strands, and their specifics make this method less usable.

6.2. Hair-optimized twisting

We discussed in Section 5.1 that hair strands do not bend over the minor axis. The theoretical way of representing this would be with an infinite bending stiffness over the minor axis. In practice, this could be represented as a very high value of the stiffness penalty μ (cf. 5.3). Unfortunately, this would result in stiff equations of motion and the Newton minimization would also become numerically unstable.

We take a different approach, keeping μ low to serve just as a correcting penalty. Then, instead of Newton minimization, we employ a different method of computing twist tailored to the fact that bending over minor axes is unwanted.

To eliminate bending over minor axes completely, the strand would have to twist so as to make the major axis parallel to the curvature binormal at each node. In our model, bending occurs at nodes, while material frames are assigned to segments. This means it is not possible to completely prevent bending over minor axes, as the nodes $\mathbf{x}_j, \mathbf{x}_{j+1}$ likely require a different direction for major axis $\mathbf{m}_j^j$. So instead, we compute twist which minimizes such bending. In effect, our twist computation is broken down into two steps:

1. Find (unoriented) direction of major axis of each segment j such that it minimizes bending over minor axes at nodes $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$. This fixes the value of θ^j up to a whole multiple of π .
2. Find the axis orientation within the direction obtained in step 1, so that it minimizes elastic energy. This determines θ^j completely by finding the multiple of π to use.

The first step is illustrated in Figure 3. It processes segments independently and so could be performed in parallel. For each segment j which is not clamped, we find the angles $\eta_{j,j+1}$ between the Bishop axis $\mathbf{u}^j$ and the respective curvature binormals at adjoining nodes, $(\kappa\mathbf{b})_{j,j+1}$. Note that as curvature binormals are perpendicular to adjoining segments, all the vectors involved are coplanar. θ^j is obtained from $\eta_{j,j+1}$ as follows:

1. If neither of $(\kappa\mathbf{b})_{j,j+1}$ is $\mathbf{0}$, $\theta^j = \frac{1}{2}(\eta_j + \eta_{j+1})$.
2. If only $(\kappa\mathbf{b})_k$ is nonzero, $\theta^j = \eta_k$.
3. If $(\kappa\mathbf{b})_{j,j+1}$ are both $\mathbf{0}$, $\theta^j = \hat{\theta}^j$.

The second step has to determine which of θ^j , $(\theta^j + \pi)$ and $(\theta^j - \pi)$ minimizes elastic energy. Let us denote energies computed with these material frames E^0 , E^+ and E^- , respectively. From the definition of $E(\Gamma)$, the following criteria can be derived:

$$\begin{aligned} E^+ < E^- &\iff \theta^j < \hat{\theta}^j \\ E^0 < E^+ &\iff -2\hat{B}^j \hat{\omega}_j^j < 2\beta\pi(\theta^j - \hat{\theta}^j) + \beta\pi^2 \\ E^0 < E^- &\iff -2\hat{B}^j \hat{\omega}_j^j < -2\beta\pi(\theta^j - \hat{\theta}^j) + \beta\pi^2, \end{aligned}$$

Figure 3: Computing material frame orientation θ^j of segment j to minimize bending over minor axes at nodes $j, j+1$. Sets θ^j to the average of (oriented) angles η_j and η_{j+1} .

where $\hat{\omega}_i^j$ stands for per-component product of ω_i^j and $\hat{\omega}_i^j$. Using these criteria, the correct value of θ^j to use can be found quickly and easily.

7. Simulation

We now assemble the equations of motion. Recall that twist is treated quasistatically and is thus not part of the dynamic simulation. So, for the equations of motion, θ^j are not independent variables, but must be expressed using $\mathbf{x}_i$.

7.1. Equations of motion

The equations of motion governing the rod's dynamic behaviour are, for $i = 0, \dots, n+1$:

$$M_i \ddot{\mathbf{x}}_i = \mathbf{F}_i^{\text{elastic}}(\mathbf{x}) + \mathbf{F}_i^{\text{external}}(\mathbf{x}, \dot{\mathbf{x}})$$

M_i is the mass of node i . $\mathbf{F}_i^{\text{elastic}}$ is the internal elastic force affecting node i ; its computation is detailed in Section 7.2. $\mathbf{F}_i^{\text{external}}$ is total external force affecting node i . In our demonstration scenarios, we use gravity and friction against static ambient air: $\mathbf{F}_i^{\text{external}} = M_i \mathbf{g} - v \dot{\mathbf{x}}_i$, where $\mathbf{g}$ is gravitational acceleration and v is air drag coefficient. The equations of motion are integrated using the symplectic Euler method [HLW06].

7.2. Elastic forces

The elastic force strives to minimize elastic energy, so we can simply write $\mathbf{F}_i^{\text{elastic}} = -\frac{dE(\Gamma)}{d\mathbf{x}_i}$. The total derivative of elastic energy takes into account both explicit dependence on centreline position and implicit dependence on it via the material frames. Therefore, to obtain an integrable formula, we must substitute into the total derivative:

$$-\frac{dE(\Gamma)}{d\mathbf{x}_i} = -\frac{\partial E(\Gamma)}{\partial \mathbf{x}_i} - \sum_{j=0}^n \frac{\partial E(\Gamma)}{\partial \theta^j} \frac{\partial \theta^j}{\partial \mathbf{x}_i}$$

We will now analyze individual components of this expression; we use the following notation: $\nabla_i = \frac{\partial}{\partial \mathbf{x}_i}$, $\nabla^j = \frac{\partial}{\partial \theta^j}$.

Holonomy To express energy derivatives, we will use the concept of *holonomy*. In our case, holonomy Ψ is a (scalar) difference of material frame rotation caused by temporal evolution of the centreline. Drawing on the notation from Sec. 6.1, we can say that ψ_i is the angle between ${}_iP(\mathbf{t}^{i-1})$ and $P_i({}_iP(\mathbf{t}^{i-1}))$. For a full description of this concept, refer to [BWR*08], from where we also take the holonomy gradient:

$$\nabla_{i-1}\Psi_i = \frac{(\mathbf{kb})_i}{2|\hat{\mathbf{e}}^{i-1}|}, \quad \nabla_{i+1}\Psi_i = -\frac{(\mathbf{kb})_i}{2|\hat{\mathbf{e}}^i|}$$

$$\nabla_i\Psi_i = -(\nabla_{i-1}\Psi_i + \nabla_{i+1}\Psi_i)$$

We can extend this concept to traversing more than one segment, as holonomy is additive. We define $\Psi^j = \sum_{i=1}^j \Psi_i$ to be the rotation needed to align ${}_iP(\dots(P_j(\mathbf{u}^0))\dots)$ to $P_1(\dots(P_j({}_iP(\mathbf{u}^0)))\dots)$. The gradient of this angle is simply $\nabla_i\Psi^j = \sum_{k=1}^j \nabla_i\Psi_k$. Notice that $\nabla_i\Psi_k$ can be nonzero for at most three values of k .

Ψ^j describes by how much the Bishop frame rotates when the centreline evolves over time. Since we define the material frame relative to the Bishop frame, we must subtract this angle to keep the material frame aligned properly. In other words, this gives us the derivative of material frame rotation with respect to centreline positions: $\frac{\partial\theta^j}{\partial\mathbf{x}_i} = -\nabla_i\Psi^j$.

Positional derivatives We now analyze $\frac{\partial E(\Gamma)}{\partial\mathbf{x}_i} = \frac{\partial E_{\text{bend}}(\Gamma) + E_{\text{twist}}(\Gamma)}{\partial\mathbf{x}_i}$. $E_{\text{twist}}(\Gamma)$ does not depend on $\mathbf{x}$ explicitly, so $\nabla_i E(\Gamma) = \nabla_i E_{\text{bend}}(\Gamma)$. The gradient takes the following form:

$$\frac{\partial E(\Gamma)}{\partial\mathbf{x}_i} = \sum_{k=1}^n \frac{1}{\hat{l}_k} \sum_{j=k-1}^k \left(\nabla_i \omega_k^j \right)^T \hat{B}^j \left(\omega_k^j - \hat{\omega}_k^j \right)$$

For its derivation and constituent terms, please refer to [BWR*08];

Twist derivatives We now turn to $\frac{\partial E(\Gamma)}{\partial\theta^j}$. Following from the relevant energy expressions, we obtain:

$$\frac{\partial E_{\text{bend}}(\Gamma)}{\partial\theta^j} = \nabla^j W_j + \nabla^j W_{j+1}$$

$$\frac{\partial E_{\text{twist}}(\Gamma)}{\partial\theta^j} = 2\beta \left(\frac{m_j}{\hat{l}_j} - \frac{m_{j+1}}{\hat{l}_{j+1}} \right),$$

where $\nabla^j W_i = \frac{1}{\hat{l}_i} (\omega_i^j)^T \hat{J} \hat{B}^j (\omega_i^j - \hat{\omega}_i^j)$

Total elastic force Recall from Section 6 that twist on unclamped segments is computed to minimize elastic energy. Therefore, $\nabla^j E(\Gamma) = 0$ for $j \notin C$. This means that the total elastic force acting on node i is:

$$\mathbf{F}_i^{\text{elastic}} = -\frac{\partial E(\Gamma)}{\partial\mathbf{x}_i} + \sum_{j \in C} \frac{\partial E(\Gamma)}{\partial\theta^j} \nabla_i \Psi^j$$

8. Constraints & collisions

As stated in Section 3, our integration scheme involves no mechanism to maintain inextensibility, avoiding equation stiffness. In [BWR*08], a post-integration constraint enforcement step is used for inextensibility and coupling of the rod to rigid bodies. We take this idea further by also using constraints to efficiently handle hair-head collisions.

8.1. Constraint types

Our model can use any number of the following constraints. Each of the constraints is represented by a value which is 0 when the constraint is satisfied; these are gathered into a constraint vector C .

Inextensibility For each segment, we define an inextensibility constraint $CI^j = \mathbf{e}^j \cdot \mathbf{e}^j - \hat{\mathbf{e}}^j \cdot \hat{\mathbf{e}}^j$.

Rigid-body coupling Any node or segment can be coupled (attached) to a rigid body. In our hair scenario, this is only used for coupling the root segment to the head. This allows us to express the coupling constraints in a simplified form: $\mathbf{CR}_0 = \hat{\mathbf{x}}_0 - \mathbf{x}_0$, $\mathbf{CR}_1 = \hat{\mathbf{x}}_1 - \mathbf{x}_1$.

Hair-head collisions After the integration step, all nodes are tested for interpenetration with the head. Penetrating nodes are gathered into set P and subjected to constraint $CH_i = (\mathbf{x}_i - \mathbf{h}) \cdot (\mathbf{x}_i - \mathbf{h}) - \hat{r}^2$ for $i \in P$, where $\mathbf{h}$ is head centre and r is head radius. In theory, nodes originally not in P could collide with the head because of relocations during the constraint enforcement. We neglect this for performance reasons; such colliding nodes will be detected and treated in the next time step. Thanks to the time step length of just 1ms, such slight interpenetrations are not noticeable.

8.2. Constraint enforcement

The constraints are enforced by *fast manifold projection* [GHF*07]. This method takes an unconstrained configuration and iteratively computes a nearby configuration which satisfies the constraints. We define “nearby” in terms of kinetic energy $\frac{1}{2} \mathbf{y} \tilde{M} \mathbf{y}^T$, where $\tilde{M}$ is a $3(n+3) \times 3(n+3)$ diagonal mass matrix and $\mathbf{y} \in \mathcal{R}^{3(n+3)}$ is system velocity:

$$\text{diag}(\tilde{M}) = (\mathbf{M}_H, \mathbf{M}_0, \dots, \mathbf{M}_{n+1}) \quad \mathbf{y} = (\dot{\mathbf{h}}, \dot{\mathbf{x}})^T,$$

where M_H is head mass and M_i is mass of node i .

After the iteration converges, we update node velocities, as per [GHF*07]: $\dot{\mathbf{x}}_i \leftarrow \dot{\mathbf{x}}_i - \frac{1}{h} \Delta \mathbf{x}_i$. Note that because of negligible hair mass compared to head mass, we do not consider moving the head in response to hair node relocations. The projection method repositions nodes only slightly. This is especially useful for our hair-head collision treatment, as the output does not exhibit unstable behaviour, often caused by use of penalty forces.

Figure 4: Snapshots from our simulation (“Stand” scenario)

Scenario	#leaders	#strands	#nodes	Time
Stand #1 long	3	72	90	1.10
Stand #2 short	3	72	15	0.13
Stand #3 short	18	450	90	0.52
Head #1 straight	50	5000	1500	24.49
Head #2 straight	25	2500	750	12.93
Head #3 wavy	25	2500	750	13.27

Table 1: Test scenario performance; time in milliseconds

9. Results and conclusion

9.1. Test scenarios

We have tested our model in the following scenarios. The first scene is a straightforward setup to validate simulation results. It consists of several wisps of hair attached to a stand with which the user can interact. Hair of different length (5cm and 30cm) was used to verify the model’s independence on hair length. Screenshots from the animation can be seen in Figure 4. This scenario was also used for twist performance comparisons, see Section 9.2.

The second scenario is a full head of hair containing in total 50 leader strands and 5000 followers. Results can be seen in Figure 5. The simulation does not reach real-time performance for this amount of nodes, but still provides a decent update frequency. We also tried the same scenario with half the number of strands, and also with wavy strands. Performance results confirm that the model is linear in the number of nodes (and hence strands).

9.2. Performance

The tests were carried out on a system with Intel® Core™2 Quad 2.66GHz CPU with 8GB of RAM. Table 1 lists the performance of our method during test scenarios. The timings given are an average for one complete simulation step. Mesh update is omitted, as it takes place at the (much lower) visual rendering frequency, not at simulation rates. We also did a comparison of our hair-specific twist computation against the original Newton iteration presented in [BWR*08]. The results are summarized in Table 2; only the time for twist computation is given.

Another test was performed to analyze the computation time taken by our hair-head collision resolution scheme. Table 3 lists the time required for constraint enforcement, both

#leaders	#nodes	Newton twisting	Our twisting
1	30	0.14	0.06
3	90	0.41	0.18

Table 2: Comparison of our method and Newton minimization, time given in milliseconds

Scenario	#leaders	#nodes	Collisions?	Time
Head #1 straight	50	1500	Yes	14.35
			No	13.54
Head #2 straight	25	750	Yes	7.87
			No	7.03
Head #3 wavy	25	750	Yes	7.97
			No	7.08

Table 3: Performance impact of hair-head collision treatment, constraint enforcement time given in milliseconds

with and without the hair-head collision constraints. These numbers clearly show that our collision handling requires just negligible overhead, while being very robust.

9.3. Conclusion and future work

We have presented a dynamic simulation model specifically tailored to simulating hair in a real-time context. We utilize specifics of hair strands to increase the stability and performance of a recently introduced simulation method [BWR*08]. We also extended our method to handle hair-head collisions within a framework already present in the simulation, thus solving the collisions in a robust way and with just minimal computational overhead.

For modest model sizes, our system provides real-time performance with rapid simulation state updates at up to 1kHz frequency. This makes the system particularly well suited to scenarios where the amount of concurrently simulated primitives can be kept low. A typical example is user interaction with hair, where simulation can be locally limited to areas the user is currently interacting with.

Another possible application is to couple our system with a global, less precise simulation method into a simulation LOD scheme. The system would coarsely simulate the entire hairstyle and use our method to obtain better accuracy in a smaller region of interest (as defined e.g. by importance).

The high update frequency offered by our method also makes it ideal for use in combination with a haptic device. We are currently investigating this area of application, aiming to design a 3D user interface enabling intuitive, yet accurate hairstyling. Such an application would present an efficient tool for 3D artists creating virtual hairstyles.

Another direction worth exploring would be to utilize the very parallel nature of the model for implementation on one of the recently emerged parallel hardware architectures such as fully programmable GPUs.

Figure 5: Snapshots from a full head simulation; (a) 5000 straight strands, (b) 2500 straight strands, (c) 2500 wavy strands

Acknowledgements The first author is supported by GA UK (Grant Agency of the Charles University), project 100209. This research is further funded by the Swiss National Science Foundation (SNSF).

References

- [BAC*06] BERTAILS F., AUDOLY B., CANI M.-P., QUERLEUX B., LEROY F., LÉVÊQUE J.-L.: Super-helices for predicting the dynamics of natural hair. In *ACM SIGGRAPH 2006 Papers* (2006), ACM Press.
- [BCN03] BANDO Y., CHEN B.-Y., NISHITA T.: Animating hair with loosely connected particles. *Computer Graphics Forum (Eurographics 2003 Proc.)* 22 (2003).
- [BK08] BONANNI U., KMOCH P.: Virtual Hair Handle: A Model for Haptic Hairstyling. In *Eurographics 2008 — Short Papers* (2008), Eurographics Association.
- [BWR*08] BERGOU M., WARDETSKY M., ROBINSON S., AUDOLY B., GRINSPUN E.: Discrete elastic rods. *ACM Trans. Graph.* 27, 3 (2008).
- [CCK05] CHOE B., CHOI M. G., KO H.-S.: Simulating complex hair with robust collision handling. In *SCA '05* (2005), ACM Press.
- [GHF*07] GOLDENTHAL R., HARMON D., FATTAL R., BERCOVIER M., GRINSPUN E.: Efficient simulation of inextensible cloth. *ACM Trans. Graph.* 26, 3 (2007).
- [Had06] HADAP S.: Oriented strands: dynamics of stiff multi-body system. In *SCA '06* (2006), Eurographics Association.
- [HLW06] HAIRER E., LUBICH C., WANNER G.: *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Series in Comp. Math. Springer, 2006.
- [HMT01] HADAP S., MAGNENAT-THALMANN N.: Modeling dynamic hair as a continuum. *Computer Graphics Forum* 20 (9 2001).
- [Pai02] PAI D. K.: STRANDS: Interactive simulation of thin solids using cosserat models. *Comp. Graphics Forum* 21, 3 (2002). Proc. of Eurographics'02.
- [SLF08] SELLE A., LENTINE M., FEDKIW R.: A mass spring model for hair simulation. *ACM Trans. Graph.* 27, 3 (2008).
- [ST07] SPILLMANN J., TESCHNER M.: CoRdE: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *SCA '07* (2007), Eurographics Association.
- [Swi95] SWIFT J.: Some simple theoretical considerations on the bending stiffness of human hair. *J. Cosmet. Sci.* 17, 6 (1995).
- [VMT04] VOLINO P., MAGNENAT-THALMANN N.: Animating complex hairstyles in real-time. In *VRST '04* (2004), ACM Press.
- [WGL04] WARD K., GALOPPO N., LIN M. C.: Modeling hair influenced by water and styling products. In *Proc. of Int. Conf. on Computer Animation and Social Agents (CASA)* (5 2004).
- [WGL06] WARD K., GALOPPO N., LIN M. C.: A simulation-based VR system for interactive hairstyling. *IEEE Virtual Reality Conference (VR 2006)* 00 (2006).
- [WL03] WARD K., LIN M. C.: Adaptive grouping and subdivision for simulating hair dynamics. In *Proc. of Pacific Graphics* (2003), IEEE Computer Society.