Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2010)
J. Bender, K. Erleben, and M. Teschner (Editors)

Interactive Simulation of a Continuum Mechanics based Torsional Thread

Karl Larsson[1], Göran Wallgren[2], Mats G. Larson[1]

[1]Umeå University, Umeå, Sweden
[2]Surgical Science, Göteborg, Sweden

Abstract

This paper introduces a continuum mechanics based thread model for use in real-time simulation. The model includes both rotary inertia, shear deformation and torsion. It is based on a three-dimensional beam model, using a corotational approach for interactive simulation speeds as well as adaptive mesh resolution to maintain accuracy. Desirable aspects of this model from a numerical and implementation point of view include a true constant and symmetric mass matrix, a symmetric and easily evaluated tangent stiffness matrix, and easy implementation of time-stepping algorithms. From a modeling perspective interesting features are deformation of the thread cross section and the use of arbitrary cross sections without performance penalty.

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

1. Introduction

In the field of physically based simulation there is a need to simulate one-dimensional, two-dimensional and three dimensional objects in a physically plausible way. Real-time simulation of such objects are of interest in games, and various types of virtual simulations such as surgery simulation for instance. This paper deals with one-dimensional objects such as beams, strands or threads.

In recent years the study of soft body simulations in the field of real-time computer graphics has seen a move away from ad-hoc simulation methods, typically based on systems of connected masses and springs, towards methods derived from the theory of continuum mechanics, typically finite element based methods. While continuum mechanics based methods normally require more computational power there are several benefits to this approach. Firstly, modeling becomes easier as measurable material parameters naturally appear in the continuum mechanics based methods compared to tweaking dozens or hundreds of masses and spring constants to produce physically plausible results. Secondly, a simulation method based on a mathematically sound theory will give greater confidence in the method and a clearer understanding of its properties and limitations based on the assumptions and approximations made in the derivation of the method.

Usually a thread is viewed as being an one dimensional object existing in three dimensional space which introduces the need for rotational variables to accurately describe the current state. This is because torsional twist cannot be described in terms of the thread midline alone. A difficulty in the dynamic simulation of threads connected to the use of rotational coordinates is that the description of the deformation field becomes highly nonlinear, thus resulting in a nonlinear mass matrix. Therefore, these rotational variables are often ignored in the dynamical representation making the model unable to capture inertial effects. Another approach is to use a simple intuitive model for the dynamics of the thread creating a need for special solution procedures to make sure the dynamical representation abide Newton's laws of motion.

In this paper we propose a thread model without rotational variables which still includes torsion and rotary inertia. A special set of three dimensional basis functions suitable for representing slender objects is used. These basis functions allow twisting along the element's principal axis. As this model is a continuum rather than a truly one-dimensional object, the simulation will in a natural way include both tor-

© The Eurographics Association 2010.

DOI: 10.2312/PE/vriphys/vriphys10/049-058

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

sion and rotary inertia. The proposed model has the following appealing features:

- Arbitrary and deformable cross section.
- Handles torsional and inertial effects.
- Symmetric tangent stiffness matrix.
- Constant and symmetric mass matrix.
- No special time stepping required.

These features make the model suitable for inclusion in an existing corotational simulation framework. As all matrices are symmetric, fast iterative solvers based on the conjugate gradient method may be used.

1.1. Our Contribution

Starting from continuum mechanics we present a straight forward derivation of a thread model for use in real-time simulation. The derivation is consistent in both the elastic description as well as the dynamic description, i.e. both the stiffness matrix and mass matrix are derived from the same deformation field.

We have applied a corotational procedure to an existing three dimensional beam element to achieve real-time simulation speeds. To our knowledge this is the first time this beam element is presented in the field of computer graphics. By using a binary tree mesh representation we allow on the fly adaptive resolution to ensure the accuracy with minimal performance impact.

2. Related Work

In recent years a number of methods for simulating torsional threads suitable for real-time simulations have been presented. Most existing methods have different representations for dynamics and elasticity, or are heavily dependent on constraint equations, making it necessary to use special solution procedures. While there are many contributions to thread simulations worth acknowledging, we limit the review of related works to thread models which include torsional effect.

A thread model which include torsion was presented by Wang et al. [WBD*05] who described the thread as a chain of springs linked at the nodes using torsional springs. The mass of the thread was assumed to be lumped at the nodes. Kubiak et al. [KPGF07] used a similar mass-spring approach in combination the ideas in [MHHR07]. With the ambitious goal of simulating every hair on a human head Selle et al. [SLF08] proposed a mass-spring model where 'altitude springs' were introduced used to capture torsional effects.

Several methods start out from the dynamic representation by modeling each thread section as a rigid body. The elastic abilities are achieved through constraints or springs acting on bending and torsional angles. Choe et al. [CCK05] proposed such a method for the modeling of hair strands and Servin and Lacoursière [SL08] proposed a similar model for simulating cables under heavy loads. A rigid body formulation was also used by Hadap [Had06] who used a differential algebraic equation solver to solve a constrained multi-body system.

The use of Cosserat rod theory for simulation of torsional threads was introduced in the field of computer graphics by Pai [Pai02] and several models based on this theory have been presented since. Grégoire and Schömer [GS07] presented a torsional thread based on Cosserat rod theory, but used it in a quasi-static approach. A method similar to [GS07] in the elastic description was presented by Spillmann and Teschner [ST07]. In their torsional thread the dynamic representation was lumped masses at each node in combination with a modified solution method to make the lumped masses follow the equations of motion for rigid bodies. The method was further refined in [ST09] where they created nets from Cosserat rods.

Bergou et al. [BWR*08] presented a torsional thread based on discrete differential geometry. However, in this model dynamical updates were only performed on the thread midline and thus did not include inertial effects. This model was further developed in the context of viscous threads by Bergou et al. [BAV*10].

Other recent work dealing with torsional threads, include the work on the super-helix model by Bertails [Ber09] and the work on creating a unified treatment for elastic rods, shells and solids by Martin et al. [MKB*10].

3. Basis Functions and Nodal Variables

Our thread model is based on a beam element presented by Shabana and Yakoub in [SY01, YS01] using a formulation called the Absolute Nodal Coordinate Formulation (ANCF). This element is continuum mechanics based and thus we use it in solving the usual equations of three dimensional elasticity. It is a two-noded element and its shape is determined by the following degrees of freedom

$$\mathbf{e} = \begin{bmatrix} \mathbf{x}_1^T & \mathbf{x}_{1,x}^T & \mathbf{x}_{1,y}^T & \mathbf{x}_{1,z}^T & \mathbf{x}_2^T & \mathbf{x}_{2,x}^T & \mathbf{x}_{2,y}^T & \mathbf{x}_{2,z}^T \end{bmatrix}^T \tag{1}$$

where $\mathbf{x}_j$ is the current spatial coordinate of node j, and $\mathbf{x}_{j,k}$ is the derivative of $\mathbf{x}_j$ with respect to $k = \{x, y, z\}$. Thus, at each node we store the position and the gradient of the deformation field. These degrees of freedom are called 'nodal coordinates'.

Given the specific basis functions for this element, the gradient variables have a geometric interpretation. The tangent vector of the beam midline is defined by $\mathbf{x}_{j,x}$ and its length defines the curvature. Because $\mathbf{x}_{j,x}$ is defined in global coordinates and shared between elements, this ensures C^1 continuity of the midline. The area and shearing of the beam cross section is defined by $\mathbf{x}_{j,y}$ and $\mathbf{x}_{j,z}$. This is illustrated in figure 1.

© The Eurographics Association 2010.

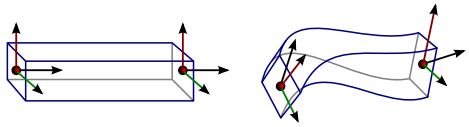

Figure 1: *Illustration of an ANCF-beam element in the reference state (left) and in a deformed state (right). The triads at each node represent the gradient of the deformation field at that node.*

In accordance with the notation for the nodal variables we let $\mathbf{x}$ be the current position of a point corresponding to a point $\mathbf{x}_r$ in the reference domain. We call $\mathbf{x}$ the deformation field. The reference domain of an element is denoted by Ω_r and the domain of the element in its rest state is denoted by Ω_e. For clarity of presentation we assume that $\Omega_e = \Omega_r$, i.e. the rest state is straight. The case of initially curved threads, where $\Omega_e \neq \Omega_r$, is addressed in Section 4.4.

Using the matrix of basis functions $\varphi(\mathbf{x}_r)$ we can write the deformation field of an element as

$$\mathbf{x} = \varphi(\mathbf{x}_r)\mathbf{e} \tag{2}$$

where $\mathbf{e}$ is the vector of nodal variables. The matrix of basis functions is

$$\varphi(\mathbf{x}_r) = \begin{bmatrix} \varphi_1\mathbf{I} & \varphi_2\mathbf{I} & \dots & \varphi_8\mathbf{I} \end{bmatrix} \tag{3}$$

with the basis functions given in [Sha08]

$$\left.\begin{array}{ll} \varphi_1 = 1-3\xi^2+2\xi^3 & \varphi_2 = l(\xi-2\xi^2+\xi^3) \\ \varphi_3 = l(\eta-\xi\eta) & \varphi_4 = l(\varsigma-\xi\varsigma) \\ \varphi_5 = 3\xi^2-2\xi^3 & \varphi_6 = l(-\xi^2+\xi^3) \\ \varphi_7 = l\xi\eta & \varphi_8 = l\xi\varsigma \end{array}\right\} \tag{4}$$

where $\xi = x/l$, $\eta = y/l$, $\varsigma = y/l$, and l is the length of the element in the reference state.

This means that the deformation field in polynomial form is

$$\mathbf{x} = \begin{bmatrix} a_0+a_1x+a_2x^2+a_3x^3+a_4xy+a_5xz+a_6yz+a_7xyz \\ b_0+b_1x+b_2x^2+b_3x^3+b_4xy+b_5xz+b_6yz+b_7xyz \\ c_0+c_1x+c_2x^2+c_3x^3+c_4xy+c_5xz+c_6yz+c_7xyz \end{bmatrix} \tag{5}$$

where x is the beam's principal direction and y and z are the transverse directions.

4. Elastic Response

Unlike the ANCF-beam element presented in [YS01] where a non-linear description based on the Green-Lagrange strain tensor

$$\varepsilon_{\mathrm{G}}(\mathbf{x}) = \frac{(\nabla\mathbf{x})^T\nabla\mathbf{x} - I}{2} \tag{6}$$

is used, we chose to evaluate the elastic forces based on linear elasticity in combination with corotation to allow for geometrically large deformation at interactive simulation speeds.

4.1. Large Deformations and Corotation

Within the field of computer graphics the corotational method presented in [MG04] and [HS04] has been successful in plausible real-time simulation of soft materials. The computational advantage is speedy computation of the elastic response forces and the tangent stiffness matrix. This is made possible by utilizing precomputed element stiffness matrices.

The fundamental assumption of corotational methods is that we locally only have small strains even though rotations may be arbitrarily large. This allows us to approximate the Green-Lagrange strain tensor ε_{G} with the corotational strain tensor, which is defined as the Cauchy strain tensor ε used in linear elasticity but in a local reference frame

$$\varepsilon_{\mathrm{CR}}(\mathbf{x}) = \varepsilon(\mathbf{u}_{\mathrm{loc}}) = \frac{\nabla\mathbf{u}_{\mathrm{loc}} + (\nabla\mathbf{u}_{\mathrm{loc}})^T}{2} \tag{7}$$

where $\mathbf{u}_{\mathrm{loc}}$ is the displacement field in the local reference frame.

As the corotational strain tensor is invariant to rigid body translations due to the gradient operators we define the local displacement field $\mathbf{u}_{\mathrm{loc}}$ without any translational terms

$$\mathbf{u}_{\mathrm{loc}} = R^T\mathbf{x} - R_0^T\mathbf{x}_0 \tag{8}$$

where R is the extracted element rotation and R_0 is the initial element rotation. Using the deformation field prescribed by our basis functions we write the local displacement field

$$\mathbf{u}_{\mathrm{loc}} = \varphi(\mathbf{x}_r)\mathbf{Q}^T\mathbf{e} - \varphi(\mathbf{x}_r)\mathbf{Q}_0^T\mathbf{e}_0 = \varphi(\mathbf{x}_r)\mathbf{d}_{\mathrm{loc}} \tag{9}$$

where the local displacement nodal variables are $\mathbf{d}_{\mathrm{loc}} = \mathbf{Q}^T\mathbf{e} - \mathbf{Q}_0^T\mathbf{e}_0$ and we have block diagonal matrices $\mathbf{Q}$ and $\mathbf{Q}_0^T$ with blocks R and R_0, respectively.

4.2. Extracting Element Rotation

Because the ANCF beam element is a continuum and thus features a complete deformation gradient, we may extract each element rotation R by polar decomposition of the deformation gradient at the element midpoint as Hauth and Strasser [HS04] and in a more elaborate setting as Moita and Crisfield [MC96]. However, as the geometry of the beam element has a natural principal axis $\mathbf{r}_1$, the axis pointing through the nodes, we create a rotation matrix R with the local x-axis pointing through the nodes. We have

$$\mathbf{r}_1 = \frac{\mathbf{x}_2 - \mathbf{x}_1}{\|\mathbf{x}_2 - \mathbf{x}_1\|} \tag{10}$$

Let $\mathbf{x}_{\mathrm{m}}$ denote the deformation field at the element midpoint. Because the beam cross section is allowed to skew,

© The Eurographics Association 2010.

we should include $\mathbf{x}_{m,y}$ and $\mathbf{x}_{m,z}$ in the calculation of $\mathbf{r}_3$, but we choose to take a more simplistic approach using the assumption that the cross section will not skew noticeably and define $\mathbf{r}_3$ through

$$\mathbf{r}_3 = \frac{\mathbf{r}_1 \times \mathbf{x}_{m,y}}{\|\mathbf{x}_{m,y}\|} \tag{11}$$

Finally we complete the orthogonal triad through

$$\mathbf{r}_2 = \mathbf{r}_3 \times \mathbf{r}_1 \tag{12}$$

The triad $R = [\mathbf{r}_1 \;\; \mathbf{r}_2 \;\; \mathbf{r}_3]$ defines the element rotation.

For simplicity and stability reasons we assume that R is constant in each time-step, and therefore also constant in changes of the nodal coordinates $\mathbf{e}$. In reality R is a function of the nodal variables and a more elaborate corotational procedure could be used, like the one presented in [Cri97].

4.3. Internal Forces and Tangent Stiffness

Assuming a Saint Venant-Kirchhoff material we may write the strain energy for an element as

$$W = \frac{1}{2}\int_{\Omega_e} \varepsilon_G^T \mathbf{D} \varepsilon_G \, dV \tag{13}$$

where, for isotropic materials, $\mathbf{D}$ is the constitutive matrix given in [ZT05] and determined by the Lamé parameters. Here we have used the symmetry of the Green-Lagrange strain tensor and written it in Voigt notation $\varepsilon_G = [\varepsilon_{11} \;\; \varepsilon_{22} \;\; \varepsilon_{33} \;\; \gamma_{12} \;\; \gamma_{23} \;\; \gamma_{31}]^T$, where $\gamma_{ij} = 2\varepsilon_{ij}$.

Assuming that the strain will locally be small we approximate the Green-Lagrange strain tensor with the corotational strain tensor (6). Thus, we rewrite the strain energy as

$$W = \frac{1}{2}\int_{\Omega_e} \varepsilon_{CR}^T \mathbf{D} \varepsilon_{CR} \, dV \tag{14}$$

Recalling the usual three-dimensional (Cauchy) strain operator $\mathcal{S}$ given in [ZT05] we can express the co-rotated strain-displacement relation as

$$\varepsilon_{CR} = \mathcal{S}\mathbf{u}_{loc} = \mathcal{S}\varphi\mathbf{d}_{loc} \tag{15}$$

Introducing the matrix $\mathbf{B} = \mathcal{S}\varphi$ we can now write the elastic energy as

$$W = \frac{1}{2}\int_{\Omega_e} \mathbf{d}_{loc}^T \mathbf{B}^T \mathbf{D}\mathbf{B}\mathbf{d}_{loc} dV = \frac{1}{2}\mathbf{d}_{loc}^T \mathbf{K}_e \mathbf{d}_{loc} \tag{16}$$

where $\mathbf{K}_e$ is the usual constant element stiffness matrix in linear elasticity

$$\mathbf{K}_e = \frac{1}{2}\int_{\Omega_e} \mathbf{B}^T \mathbf{D}\mathbf{B} dV \tag{17}$$

We can simulate arbitrary cross sections through the choice of Ω_e. As the element stiffness matrix is precomputed, there is performance wise no limitation in the complexity of the cross section. In Figure 2 a simulation using an oval cross section is shown. For estimation of the integrals when using an arbitrary cross section, a triangulation of the cross section can be used. Even though this cross section may be chosen arbitrarily, it is recommended to keep the cross section center of mass at the midline defined by the nodes as interaction otherwise will appear unintuitive. Further, the basis functions may exhibit undesired behavior away from the midline.

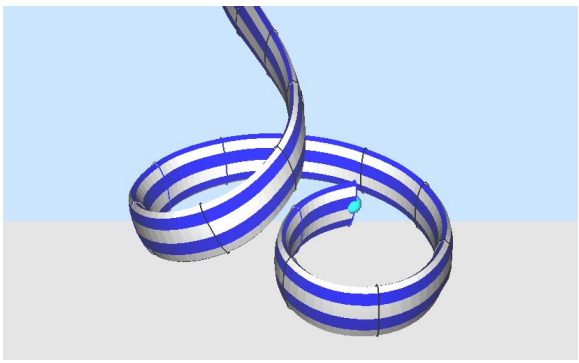

Figure 2: *Simulation with oval cross section.*

As it is assumed that R is constant between time-steps we can use the approximation

$$\frac{\partial \mathbf{d}_{loc}}{\partial \mathbf{e}} \approx \mathbf{Q}^T \tag{18}$$

We derive the internal forces vector for the element by differentiating the strain energy (13)

$$\begin{aligned} \mathbf{f}_{int} &= \left(\frac{\partial W}{\partial \mathbf{e}}\right)^T = \left(\frac{\partial W}{\partial \mathbf{d}_{loc}}\frac{\partial \mathbf{d}_{loc}}{\partial \mathbf{e}}\right)^T \\ &= \left(\frac{\partial \mathbf{u}_{loc}}{\partial \mathbf{e}}\right)^T \mathbf{K}_e \mathbf{d}_{loc} = \mathbf{Q}\mathbf{K}_e \mathbf{d}_{loc} \end{aligned} \tag{19}$$

The tangent stiffness matrix is calculated by differentiating the internal forces vector

$$\mathbf{K}_{eT} = \frac{\partial \mathbf{f}_{int}}{\partial \mathbf{e}} = \mathbf{Q}\mathbf{K}_e\mathbf{Q}^T \tag{20}$$

Note that this derivation is general and produces the same corotational method as the one presented for linear tetrahedra in [HS04], but with a different element stiffness matrix.

4.4. Initially Curved Threads

For initially curved threads we have that $\Omega_e \neq \Omega_r$. The ANCF-beam element is an isoparametric element and as such we may define the rest state using our basis functions

$$\mathbf{x}_0 = \varphi(\mathbf{x}_r)\mathbf{e}_0 \tag{21}$$

© The Eurographics Association 2010.

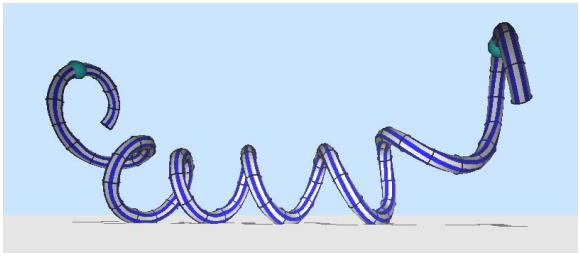

Figure 3: *Simulation of a thread with curved initial state.*

where $\mathbf{x}_0$ is a point in the rest state domain Ω_e, $\mathbf{x}_r$ is a point in the reference domain Ω_r, and $\mathbf{e}_0$ is a constant vector of nodal variables.

The displacement gradient for a curved element is

$$\nabla \mathbf{u}_{\text{loc}} = \frac{\partial \mathbf{u}_{\text{loc}}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{u}_{\text{loc}}}{\partial \mathbf{x}_r}\frac{\partial \mathbf{x}_r}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{u}_{\text{loc}}}{\partial \mathbf{x}_r} J_0^{-1} \tag{22}$$

with the matrix $J_0 = \partial \mathbf{x}_0 / \partial \mathbf{x}_r$. This will result in a modified strain operator $\mathcal{S}$ in (15), which is used in the calculation of the $\mathbf{B}$ matrix. Note that for $\mathbf{x}_0 = \mathbf{x}_r$, J_0^{-1} is the identity matrix.

By a change of variables from $\mathbf{x}_0$ to $\mathbf{x}_r$ we describe the element stiffness matrix as an integral over the reference domain Ω_r

$$\mathbf{K}_e = \frac{1}{2}\int_{\Omega_r} \mathbf{B}^T \mathbf{D}\mathbf{B} |J_0| dV \tag{23}$$

As this integral is calculated by numerical integration, we find J_0^{-1} at each quadrature point by taking the inverse of J_0.

4.5. Element Limitations

The main assumption in corotational formulations is that locally the strains are small. If this assumption is false in an element we may encounter intra-element artifacts such as those presented in Figure 4. There are two causes for these artifacts, mainly incorrect estimation of internal forces when the corotational assumption is false, but also limitations in what shapes the deformation field can take. Through refinement, these intra-element artifacts are efficiently resolved.

5. Adaptive Resolution

When deformation of a single element becomes large, strains become large and artifacts such as those presented in Section 4.5 may occur. These artifacts are resolved by splitting the element. By splitting elements before deformation becomes too large artifacts can altogether be avoided. In our implementation we created ad-hoc refinement and coarsening criteria based on bending and twisting angles. By choosing suitable constants we introduce some hysteresis between refinement and coarsening to avoid frequently alternating resolution changes.

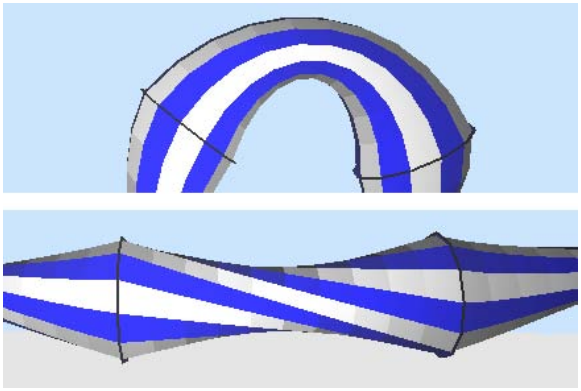

Figure 4: *Intra-element artifacts stemming from large bending (top) and large twist (bottom).*

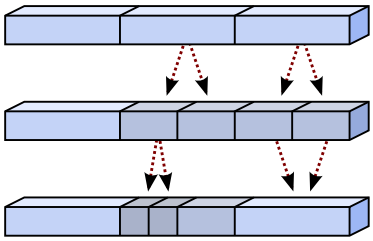

Figure 5: *Illustration of mesh refinement and coarsening using a hierarchical mesh level structure.*

Spillmann and Teschner [ST08] use an energy minimizing procedure to select the position of new nodes. This is done to remove the snapping effect which may occur after splitting an element. However the need for such a procedure is limited in the presented model. Since the deformation field is known at all points of the thread and the midline is represented as a C^1 continuous curve we get a natural choice of nodal coordinates and velocities when inserting new nodes in the thread. Thus, the deformation field after the split will be the same and also its velocities. As long as we split elements before artifacts become too large, snapping effects can be avoided. Assuming that the rest shape of initially curved threads is correctly described in the coarsest allowed state, the rest shape will remain the same after a change in resolution.

5.1. Hierarchical Representation

To be able to dynamically scale the resolution up and down without too big impact on performance, we chose to have a hierarchical representation of the mesh. This means that mesh refinement is performed by symmetric splitting of elements and coarsening is performed by joining previously split elements as illustrated in Figure 5. The implementation is done using a binary tree to represent the mesh and

© The Eurographics Association 2010.

element stiffness and mass matrices are precomputed for every refinement level to minimize the performance impact of splitting elements.

5.2. Ad-hoc Refinement/Coarsening Criteria

We constructed an ad-hoc refinement criteria by defining error indicators for an element

$$\eta_{\text{bending}} = \left| \frac{\mathbf{r}_1 \cdot \mathbf{x}_{1,x}}{\|\mathbf{x}_{1,x}\|} \right| + \left| \frac{\mathbf{r}_1 \cdot \mathbf{x}_{2,x}}{\|\mathbf{x}_{2,x}\|} \right| \tag{24}$$

$$\eta_{\text{torsion}} = \left| \frac{\mathbf{r}_2 \cdot \mathbf{x}_{1,y}}{\|\mathbf{x}_{1,y}\|} \right| + \left| \frac{\mathbf{r}_2 \cdot \mathbf{x}_{2,y}}{\|\mathbf{x}_{2,y}\|} \right| \tag{25}$$

and recommended splitting of the element if

$$\eta_{\text{refinement}} < \eta_{\text{bending}} + \eta_{\text{torsion}} \tag{26}$$

where $\eta_{\text{refinement}}$ is a constant.

The same error indicator was used to determine coarsening, with the left and right nodes taken from a pair of elements. If the error indicator is smaller than a constant

$$\eta_{\text{coarsening}} > \eta_{\text{bending}} + \eta_{\text{torsion}} \tag{27}$$

joining of the element pair is recommended.

Because of the binary tree representation of the mesh only two elements sharing the same parent in the binary tree are allowed to be joined. By only allowing one refinement and one coarsening per timestep the computational cost of refinement/coarsening is limited and so are any noticeable side effect of refinement/coarsening.

These criteria were applied to an initially straight thread, but can also be used on initially curved threads by transforming the nodal gradients to the reference state by multiplying them with J_0^{-1}.

6. Dynamics

In the presented method there is no special treatment of the mass matrix as it is directly derived from the deformation field, and still it becomes constant and capture inertial effects.

We can write the kinetic energy by integrating the displacement field over the domain of the undeformed element.

$$T = \frac{1}{2} \int_{\Omega_e} \rho \dot{\mathbf{x}}^T \dot{\mathbf{x}} dV \tag{28}$$

As the deformation field for each element can be represented as $\mathbf{x} = \varphi(\mathbf{x}_r)\mathbf{e}$ with our time-dependent nodal variables as coefficients we can easily derive the constant element mass matrix $\mathbf{M}_e$ via

$$T = \frac{1}{2}\dot{\mathbf{e}}^T \left[\int_{\Omega_e} \rho \varphi(\mathbf{x}_r)^T \varphi(\mathbf{x}_r) dV \right] \dot{\mathbf{e}} = \frac{1}{2}\dot{\mathbf{e}}^T \mathbf{M}_e \dot{\mathbf{e}} \tag{29}$$

That this constant mass matrix correctly handles inertial effects is shown in [YS01].

6.1. Time Stepping

The same degrees of freedom are used in the calculation of the dynamic response as in the elastic response, and no constraints are introduced. It is therefore straight forward to apply a finite difference scheme to produce a time stepping algorithm.

Let $\mathbf{M}$, $\mathbf{K}_T$ and $\mathbf{F}^{\text{int}}$ denote global matrices and vectors which, as usual, are assembled by summing element contributions from each element. The state vector $\mathbf{e}$ consists of nodal variables for all nodes and the equations of motion take the form

$$\mathbf{M}\ddot{\mathbf{e}} = \mathbf{F}^{\text{int}} + \mathbf{F}^{\text{ext}} \tag{30}$$

We rewrite this differential equation as a system of first order ODEs:

$$\begin{cases} \dot{\mathbf{e}} = \mathbf{v} \\ \mathbf{M}\dot{\mathbf{v}} = \mathbf{F}^{\text{int}} + \mathbf{F}^{\text{ext}} \end{cases} \tag{31}$$

We choose to discretize time with a time step h using the backward Euler method, which gives

$$\begin{cases} \dfrac{\mathbf{e}_n - \mathbf{e}_{n-1}}{h} = \mathbf{v}_n \\ \mathbf{M}\dfrac{\mathbf{v}_n - \mathbf{v}_{n-1}}{h} = \mathbf{F}_n^{\text{int}} + \mathbf{F}_n^{\text{ext}} \end{cases} \tag{32}$$

Our best approximation for the external forces at the next time-step n are the forces currently applied. For the internal forces we make a better approximation by doing a single Taylor expansion of $\mathbf{f}_{\text{int}}$ around $\mathbf{e}_{n-1}$.

$$\mathbf{F}_n^{\text{ext}} \approx \mathbf{F}_{n-1}^{\text{ext}} \tag{33}$$

$$\mathbf{F}^{\text{int},n} \approx \mathbf{F}_{n-1}^{\text{int}} + \left. \frac{\partial \mathbf{f}_{\text{int}}}{\partial \mathbf{e}} \right|_{n-1} (\mathbf{e}_n - \mathbf{e}_{n-1}) \tag{34}$$

From (20) we obtain

$$\mathbf{F}_n^{\text{int}} \approx \mathbf{F}_{n-1}^{\text{int}} + \mathbf{K}_{T,n-1}(\mathbf{e}_n - \mathbf{e}_{n-1}) \tag{35}$$

The final time stepping algorithm is derived by solving for $\mathbf{e}_n$ and $\mathbf{v}_n$ in the system

$$\begin{cases} \dfrac{\mathbf{e}_n - \mathbf{e}_{n-1}}{h} = \mathbf{v}_n \\ \mathbf{M}\dfrac{\mathbf{v}_n - \mathbf{v}_{n-1}}{h} = \mathbf{K}_T(\mathbf{e}_n - \mathbf{e}_{n-1}) + \mathbf{F}_{n-1}^{\text{int}} + \mathbf{F}_{n-1}^{\text{ext}} \end{cases} \tag{36}$$

Note that this presented method only involves a single Newton step which should be sufficient given that the time step is small enough. The same time integration in a slightly different setting is presented in [BW98].

6.2. Collision handling

When interacting with a real-world thread it is likely to collide with both the surrounding environment and itself. Therefore, besides time-stepping the internal dynamics of our torsional thread, we also need to handle collisions and

© The Eurographics Association 2010.

contact situations robustly and efficiently to make for a plausible and interesting real-time simulation. In fact, when simulating the twisting stiffness of a torsional thread it will self-collide more frequently than a simpler model that neglects the torsional effects. This is due to the tendency of a twisted thread to twirl itself up when given some slack, see Figure 6.

It is beyond the scope of this paper to give a detailed description of the intricacies of collision detection and response, but we will give a quick overview below. For an excellent further read on the topic, see the State of The Art Report by Teschner et al. [TKZ*05]. Our implementation is mainly based on ideas from the cloth collision treatment described by Bridson et al. [Bri02].

For the purpose of producing the examples in this paper, we adopted a simple collision geometry. Each element in our thread model was approximated with a collision volume consisting of a line segment between the nodes $\mathbf{x}_1$ and $\mathbf{x}_2$, padded with a thickness matching the average radius of the element cross section. This is equivalent to a cylinder joined with half-sphere caps on both ends, also called a capsule. The penetration depth of two such colliding capped cylinders can be calculated by comparing the sum of their radii against the shortest distance between the center-line segments. For collisions of the thread against an infinite ground plane, we simplify things even further by checking the signed distance between each node $\mathbf{x}_j$ and the plane. It should be noted, however, that the approximate collision volumes described here may not be accurate enough if the element cross section is not circular or if a single element is allowed to bend significantly. In that case, a better fit could be achieved by subdividing these collision volumes into smaller parts or even triangulating the thread surface and using a more complicated triangle-mesh based approach.

For a thread with just a few segments it may be sufficient to do a brute-force collision detection by testing all thread segments against all the others, yielding a $O(n^2)$ algorithm. However, for practical purposes the collision detection performance should be accelerated by efficiently pruning, or culling, unnecessary collision tests using a spatial data structure. For deformable geometry like our thread model, the chosen data structure must also be able to handle frequent updates. Two good choices in our case would be either a spatial hashing algorithm or a bounding volume hierarchy, where the latter could be built in accordance with the binary tree used for adaptivity in Section 5.1. An efficient bounding volume for thread segments is an extension of the axis aligned bounding box (AABB) where all edges are beveled. This is also known as a k-DOP (Discrete Oriented Polytope) where the number of half-spaces used for its construction is k = 18. For the relatively short threads in our implementation, we skipped the hierarchy and tested all possible pairs of elements, excluding pairs of node-sharing neighbor elements. Each element was encapsulated in an 18-DOP for quick culling of distant element pairs.

For slow movement or short time-steps one could use discrete collision detection, i.e. testing for inter-penetration at the end of each time step. However, for interactive simulation rates the velocities of thread segments may easily become large enough to cause "tunneling" where a collision is missed since the thin thread segments pass completely through each other during one single timestep. To catch those cases a more elaborate continuous collision detection method must be used, where the trajectories of thread segments is checked for collisions during a timestep. In our implementation we used dynamic intersection tests based on interval halving to avoid missing any collisions (see Chapter 5.5 in the book by Ericson [Eri04]).

When collisions are detected, appropriate responses must be applied to resolve the collisions. This can be done in several ways. Inter-penetrations can be removed directly by moving nodes towards a collision free state and modifying their velocities. In that case a strain or strain rate limiting method may need to be applied to smooth out resulting discontinuities in the neighborhood of a collision. Collision responses could also be applied indirectly by feeding them back into the dynamics solver as springs, penalty forces or constraints. For a robust handling of collisions, care must be taken to detect any new collisions caused by applied collision responses. It may sometimes be necessary to iterate the collision resolving process several times for a single time step until all collisions are resolved. For our implementation we chose to modify node positions and velocities directly, in combination with a simple strain limiting approach.

7. Numerical Examples

The purposes of these numerical examples are threefold:

- Show that the presented method capture desired properties of torsional threads.
- Demonstrate that the use of an adaptive resolution mesh produces plausible results comparable to those of a uniform fine scale mesh.
- Illustrate unique or interesting features of the proposed thread model.

The first example shown in Figure 6 is a typical case of demonstrating the torsional abilities of a thread model. Twisting the right end of the thread while the left end is locked, produces the desired effect that the thread twirls around itself. Such an example is not possible using models without torsion.

Figure 7 illustrates the use of adaptivity when the thread is wrapped around a stiff object, for instance a surgical instrument. The problem with the coarse mesh is twofold. Firstly, the elements become too stiff when bent too much. Secondly, the beam midline is C_1 continuous while the collision volumes are piece-wise linear, causing an unnatural distance to the instrument. These problems are solved when using a

© The Eurographics Association 2010.

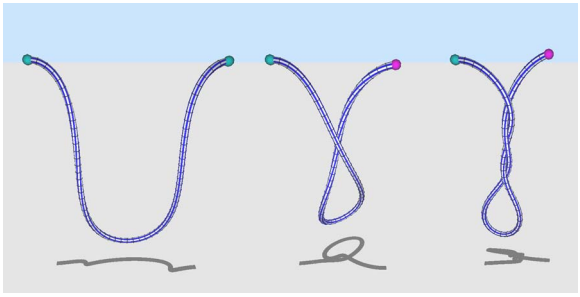

Figure 6: *Thread with increasing torque applied at the right node to produce a torsional effect.*

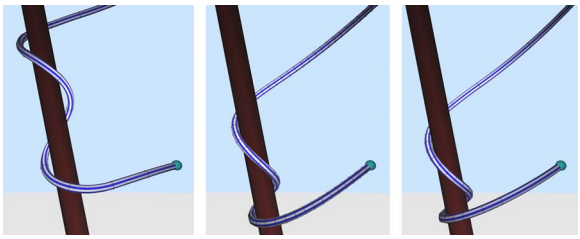

Figure 7: *Thread with various levels of mesh refinement wrapped around a fixed rod: coarse mesh (left), fine mesh (center), adaptive mesh (right).*

fine mesh or an adaptive mesh. Note that the difference between the fine mesh solution and adaptive mesh solution is minor. Another example that visualizes the use of adaptive resolution is shown in Figure 8 where we have made a knot on the thread. This is typical case where it is valuable to have a high resolution mesh in very local areas, why adaptive mesh resolution is a valuable feature.

To visualize the ability of the cross section to deform we used a soft rubber like material with somewhat volume conserving properties which is stretched in Figure 9. Note that no special procedures is used to ensure volume conservation. As no locking effects introduced by the mesh, as in the case

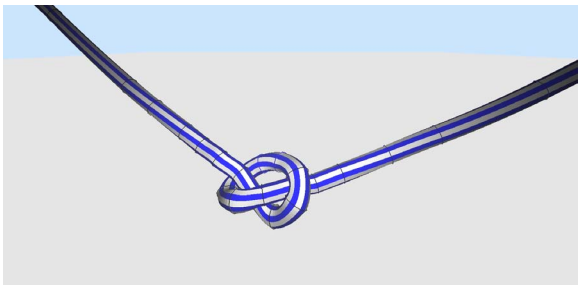

Figure 8: *Thread with a knot. The adaptive resolution automatically increase resolution where needed.*

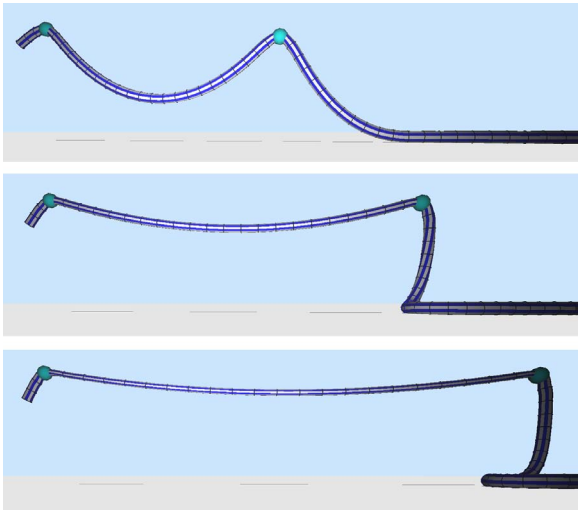

Figure 9: *Stretching a rubber like material, illustrating the deformation of the thread cross section. A Poisson's ratio of* $\nu = 0.4$ *was used in this example.*

of two and three dimensional structures, it is possible to use a Poisson's ratio close to 0.5.

Other features of this model is the support for curved rest states as visualized by the phone cord shape in Figure 3 and the support for arbitrary cross sections as visualized in Figure 2 by using a simple non-trivial cross section, an oval. As all the complexities of these features are concentrated to the calculation of the element stiffness and mass matrices, which are precalculated, there is no added performance penalty.

7.1. Visualization

In all the presented numerical examples we utilize the known deformation field in our visualization. We sample points on the surface of each element, dividing the circumference of the cross section into 12 parts and the length of the element into 10 parts. Torsional twists is illustrated by blue and white stripes along the element while the mesh is illustrated by blue rings at each nodal point.

7.2. Computational Cost

The number of degrees of freedom for the proposed thread is greater than for most existing methods. The model consists of $12(n+1)$ degrees of freedom, where n is the number of elements. For comparison the CoRdE method [ST07, ST08] consists of $7(n+1)$ degrees of freedom for a thread of n elements. It is likely that this will result in a higher computational cost for the proposed method if comparing threads using the same number of elements, of course depending on the solution method. It should be noted that the computational cost is independent of the choice of cross section and initial

© The Eurographics Association 2010.

curvature if we ignore visualization and collision aspects. The extra degrees of freedom also means that the proposed method is richer, in that the cross-section of the thread may shear and deform. This can be useful for simulating materials with volume conserving properties, such as rubber bands (See Figure 9).

The proposed thread model has a clearly defined deformation field with C_1 continuity along the principal axis. By rendering the known deformation field between element nodes we create smooth curves even for threads using few elements. However, for simulations including collision detection this creates a non-conformity between the visualization and the collision volumes because suitable collision volumes in thread simulations are cylinders.

Simulating a thread consisting of 64 elements on an Intel iCore 870 CPU a framerate of 95 Hz was achieved in our single-threaded program with collision detection disabled and visualization kept basic. While we solve the linear system of equations using a conjugate gradient method the banded sparsity pattern of the matrices allow for using more efficient solver implementations [KW03].

8. Conclusion

In this paper we have adapted the ANCF-method for continuum based beams to a corotational setting for use in real-time thread simulations. By employing an adaptive refinement and coarsening procedure we effectively remove corotation artifacts stemming from locally large deformations.

To summarize, the benefits of this approach are

- Possibility to simulate slender objects with any cross section and the ability for the cross sections to deform.
- Suitable for inclusion in existing corotational frameworks.
- Symmetric mass and stiffness matrices, thus allowing the use of fast iterative Conjugate Gradient solvers.
- Any finite difference scheme can easily be used to create the time stepping algorithm.
- A straight forward derivation from the fundamental equations of continuum mechanics.

8.1. Limitations and Future Work

For some applications within the field of computer simulations this model may be a bit too rich in that the thread cross section is allowed to deform and shear. Therefore it would be interesting to investigate the possibility to reduce the degrees of freedom in this model without reducing its desirable properties, i.e. constant mass matrix, rotary inertia and torsional effects.

As this method handles large deformations by a corotational approach, where the element rotation is assumed constant during each timestep, the method has the inherent limitations of such approaches, i.e. extreme deformations give rise to an oscillating deformation field when the time step is kept constant.

The purpose of applying a corotational approach to the ANCF-beam is to approximate the non-linear formulation [SY01, YS01] in a computationally efficient way. To estimate how good this approximation is, we would like to implement the non-linear formulation to get reference solutions to compare our results with.

References

[BAV*10] BERGOU M., AUDOLY B., VOUGA E., WARDETZKY M., GRINSPUN E.: Discrete Viscous Threads. *ACM Trans. on Graphics* (2010). 2

[Ber09] BERTAILS F.: Linear time super-helices. *Computer Graphics Forum 28*, 2 (2009), 417–426. 2

[Bri02] BRIDSON R.: Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. on Graphics 21* (2002), 594–603. 7

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH* (1998), pp. 43–54. 6

[BWR*08] BERGOU M., WARDETZKY M., ROBINSON S., AUDOLY B., GRINSPUN E.: Discrete Elastic Rods. *ACM Trans. on Graphics* (2008). 2

[CCK05] CHOE B., CHOI M. G., KO H.-S.: Simulating complex hair with robust collision handling. In *Proc. of ACM SIGGRAPH* (2005), pp. 153–160. 2

[Cri97] CRISFIELD M. A.: *Non-linear Finite Element Analysis of Solids and Structures: Volume 2*. John Wiley & Sons, Chichester, 1997. 4

[Eri04] ERICSON C.: *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. 7

[GS07] GRÉGOIRE M., SCHÖMER E.: Interactive simulation of one-dimensional flexible parts. *Comput. Aided Des. 39*, 8 (2007), 694–707. 2

[Had06] HADAP S.: Oriented strands: dynamics of stiff multibody system. In *Proc. of ACM SIGGRAPH* (2006), pp. 91–100. 2

[HS04] HAUTH M., STRASSER W.: Corotational simulation of deformable solids. In *Proc. WSCG* (2004), pp. 137–145. 3, 4

[KPGF07] KUBIAK B., PIETRONI N., GANOVELLI F., FRATARCANGELI M.: A robust method for real-time thread simulation. In *Proc. ACM symposium on Virtual reality software and technology* (2007), pp. 85–88. 2

[KW03] KRÜGER J., WESTERMANN R.: Linear algebra operators for GPU implementation of numerical algorithms. *ACM Trans. on Graphics 22*, 3 (2003), 908–916. 9

[MC96] MOITA G. F., CRISFIELD M. A.: A finite element formulation for 3-d continua using the corotational technique. *Int. J. for Numerical Methods in Eng. 39* (1996), 3775–3792. 3

[MG04] MÜLLER M., GROSS M.: Interactive virtual materials. In *Proc. Graphics Interface* (2004), pp. 239–246. 3

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *J. Vis. Comun. Image Represent. 18*, 2 (2007), 109–118. 2

[MKB*10] MARTIN S., KAUFMANN P., BOTSCH M., GRINSPUN E., GROSS M.: Unified simulation of elastic rods, shells, and solids. *ACM Trans. Graphics 29*, 4 (2010), 1–10. 2

© The Eurographics Association 2010.

[Pai02] PAI D. K.: Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum 21*, 3 (2002), 347–352. 2

[Sha08] SHABANA A. A.: *Computational Continuum Mechanics*. Cambridge University Press, 2008, ch. 6, p. 271. 3

[SL08] SERVIN M., LACOURSIÈRE C.: Rigid body cable for virtual environments. *IEEE Transactions on Visualization and Computer Graphics* (2008). 2

[SLF08] SELLE A., LENTINE M. G., FEDKIW R.: A mass spring model for hair simulation. *ACM Trans. on Graphics 27*, 3 (2008). 2

[ST07] SPILLMANN J., TESCHNER M.: Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *Proc. ACM SIGGRAPH* (2007), pp. 63–72. 2, 8

[ST08] SPILLMANN J., TESCHNER M.: An adaptive contact model for the robust simulation of knots. *Computer Graphics Forum 27*, 2 (2008), 497–506. 5, 8

[ST09] SPILLMANN J., TESCHNER M.: Cosserat nets. *IEEE Transactions on Visualization and Computer Graphics 15*, 2 (2009), 325–338. 2

[SY01] SHABANA A. A., YAKOUB R. Y.: Three dimensional absolute nodal coordinate formulation for beam elements: Theory. *J. of Mechanical Design 123* (2001), 606–613. 2, 9

[TKZ*05] TESCHNER M., KIMMERLE S., ZACHMANN G., HEIDELBERGER B., RAGHUPATHI L., FUHRMANN A., CANI M.-P., FAURE F., MAGNETAT-THALMANN N., STRASSER W.: Collision detection for deformable objects. *Computer Graphics Forum 24*, 1 (2005), 61–81. 7

[WBD*05] WANG F., BURDET E., DHANIK A., POSTON T., TEO C. L.: Dynamic thread for real-time knot-tying. In *Proc. Eurohaptics Conf. and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (2005), pp. 507–508. 2

[YS01] YAKOUB R. Y., SHABANA A. A.: Three dimensional absolute nodal coordinate formulation for beam elements: Implementation and applications. *J. of Mechanical Design 123* (2001), 614–621. 2, 3, 6, 9

[ZT05] ZIENKIEWICZ O. C., TAYLOR R. L.: *The Finite Element Method for Solid and Structural Mechanics*, 6 ed. Butterworth-Heinemann, 2005. 4

© The Eurographics Association 2010.