MICADO: Models of Interactive Constraints for the Assembling of 1D Deformable Objects

Adrien Theetten, Laurent Grisoni

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

MICADO: Models of Interactive Constraints for the Assembling of 1D Deformable Objects

A. Theetten — L. Grisoni

N° 6573

July 2008

Thème COG

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--6573--FR+ENG

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
INRIA
centre de recherche LILLE - NORD EUROPE

MICADO: Models of Interactive Constraints for the Assembling of 1D Deformable Objects

A. Theetten* , L. Grisoni *

Thème COG — Systèmes cognitifs

Équipe-Projet Alcove

Rapport de recherche n° 6573 — July 2008 — 23 pages

Abstract: This paper introduces a set of Lagrangian constraints, allowing most needed interaction and combinations of one-dimensional deformable elements for creating complex structures. The proposed tools can potentially be used with a large set of available 1D-models. All constraints formulation are compatible with linear, displacement-based, integration schemes. The proposed constraints allow for real-time complex structure simulation, and also novel interactions between simulated objects. Various examples are provided, illustrating the benefit of the proposed numerical tools.

Key-words: physical simulation, virtual reality, 1D deformable models, lagrangian constraints

* contact author: laurent.grisoni@lifl.fr

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne
40, avenue Halley, 59650 Villeneuve d'Ascq
Téléphone : +33 3 59 57 78 00 — Télécopie : +33 3 59 57 78 50

Modèles de Contraintes Interactives pour l'Assemblage de Modèles Dèformables 1D

Résumé : Cet article décrit un ensemble de contraintes lagrangiennes, permettant de modéliser la plupart des interactions nécessaires, et combinaisons, de modèles déformables 1D pour créer des structures complexes. Les outils proposés sont compatibles avec un large ensemble de modèle 1D disponibles. Toutes les formulations de contraintes sont compatibles avec les schémas d'intégration implicites, basés position. Cet ensemble permet une simulation temps réel de structures complexes, ainsi que de nouvelles interactions entre objets simulés. Des exemples sont fournis, illustrant le bénéfice de ces outils numériques.

Mots-clés : simulation physique, réalité virtuelle, modèles déformables 1D, contraintes lagrangienne

Figure 1: A bridge modeled with 4 deformable ropes, linked with distance constraints. Such constraints improve efficiency, maintaining realistic visual behavior.

Figure 2: Threading of a pearl necklace, illustrating the interaction between the 1D deformable model and the pearls.

1 Introduction

In recent years thin elastic solids have become more and more used in the computer graphics community. An important research effort has been achieved on modeling such objects. It is now possible to handle deformations in an interactive, and mechanically accurate way. Such models can be used for modeling objects that obviously look like deformable lines (such as ropes [ST07], surgical threads [CDL+05], cables [GS07],etc.). They can also be used as an animation skeleton for more complex shapes (e.g. hair wisps [BAC+06], muscles...). Surprisingly, while the community has made a significant research effort on the mechanical models for 1D-objects, only few work has been done on the methods and tools for sophisticated interaction on these objects. Equally, numerical tools for tight coupling between such models, in order to create complex structures, have been somewhat ignored. Lagrangian constraints are known to be an efficient method for linking models together. They can also produce effects that classical collision-based methods can barely provide (e.g. static link between objects). However, it is difficult to find non-trivial examples of such constraints in the computer graphics literature. Research on one-dimensional deformable models is probably still a hot topic in the field; such models are currently mature enough in the animation community for next step, that is combining them together.

Some Lagrangian constraints have already been proposed for one-dimensional deformable models: position, plane, axis as well as *smooth constraints*, i.e. constraints that impose the deformable model to slide through a static space point [LGM+04] (see next section for further details). Most of these formulations have been made through acceleration-based equation. They demand specific additional computation. In this article we explain how to formulate these constraints for displacement-based backward integration methods [DSB99], that are known to provide efficient results for most real-time simulations, and have now become very popular. To our knowledge, there is a critical lack of non-trivial examples, in the literature, of Lagrangian constraints formulation, that are compatible with these integration schemes.

We propose here a set of constraints, that can accurately simulate important behaviors. Few of these constraints are straightforward (e.g. defining a static position on a deformable object is very easy and very common). We also propose novel constraint tools (e.g. sliding constraints). We provide here a large set of constraints for one-dimensional deformable objects, but also a complete framework that unifies these constraints. We first define *position* and *orientation constraints* for user-defined position and direction at specific parameter values; *axis*, *plane* and *distance constraints* for partially constraining the motion of a specific point. Since 1D elements are not often used alone, we also introduce *fusion* constraints for branching construction, to achieve complex structure like cable trees, or rope bridge. Finally, we propose two different *sliding constraints* for complex interaction, that are visually quite different: the first one is the *keyhole* constraint that allows the model to slide through a specified point. The second one is the *pearl* constraint, that allows objects to slide on the deformable model.

These constraints can be applied on a large set of one-dimensional deformable models. Many models for 1D-deformable objects exist in computer graphics community. In the literature, such models are called either *rods*, *strands*, *cordes*, *dynamic splines*, *super-helices*, *material splines*, *beams*... Because we make only poor assumption on the chosen model (see section 3.1), and exhibit generic methods, in this article we reference the considered model as *Deformable One-Dimensional Element*, or *DODE* for short.

The rest of this paper is organized as follows. After a short overview on related works in section 2, we describe the required assumptions on the DODE model. We provide some basic framework for Lagrangian constraints in section 3. Then we propose the set of constraints that constitutes MICADO, in the following sections. In the first of these sections, we present how to formulate basic constraints on a single DODE point (section 4). Second, we show how to combine DODEs together, introducing fusion constraints in section 5. Third, we propose two sliding constraints with totally different behavior, keyhole and pearl constraints, in section 6. Finally, we present various examples of MICADO advantages, through various interactive applications: cable bundle on a car frame,

a bridge and a spider net interactively manipulated, pearl dynamic threading, and real-time bow-arrow physical simulation (section 7).

2 Related works

The literature about DODE models has been very proficient in recent years. Proposed models usually aim at most of the following purposes: deformation accuracy and real-time performance, as well as easy handling. Recent works show that these objectives are usually antagonist in practice. To our knowledge, a model that fulfill all of them has not been proposed yet. Here, we classify them in three categories, depending on their discretization level: particle systems, chain approaches and continuous methods.

The most intuitive and easiest way to simulate a DODE is to consider punctual masses connected with springs [Mil88, PLK02], or coupling them with quaternions for better realism [GS07]. However, this approach does not offer enough accuracy and stability, especially for stiff elements.

Rigid chain approaches [Fea87, RGL05, GLM06] are well-known for their robustness and efficiency, but do not produce valid and accurate deformations, except for non real-time Hadap's dynamic stiff elements [Had06]. Constraints and external forces can properly be accounted for when using a multi-pass forward dynamics algorithm, but are not suitable for chains with high bending and torsional stiffnesses [WBK+07]. Some methods use chains of deformable elements, such as pure bending linear beams [AUK92] or as Cosserat elements [Pai02, BAC+06] which provide much more realistic deformations. However, these methods are particularly difficult and expensive to constrain in position and orientation an element between the extremities, since they use a reduced coordinate parametrization [WBK+07, ST07].

Continuous methods provide more accurate results and usually a better handling of DODEs. The Finite Element Method has been widely used in mechanics and computer graphics to simulate deformable object for many years. One dimensional FEM [SVQ88, BP05, WH04] valued high accuracy over efficiency and handling. Cotin *et al.* [CDL+05] used an incremental integration of serial linear elements to provide a non linear global behavior but their algorithm still requires a large amount of elements for accuracy. Spillmann *et al.* [ST07] recently proposed a very efficient FEM version of Gregoire's work [GS06], mixing positions and quaternion orientations, but this method is burdened with the inability of being handled between the cable extremities. Another continuous model, the dynamic spline, has been widely used for 1D animation [TPBF87] and sculpturing [QT96]. Many works improved this model to provide a mechanically valid and interactive tool for 1D simulation, first with connected springs between control mass-points [RNG99, LMGC02], then with continuous formulations of stretching, bending and twisting strains [NR01, LGMC05, TGDM07].

Some results are available in literature on Lagrangian constraints as well [BB88, WW90, RNN01]. The Lagrange multipliers are a well-known solution to impose boundary conditions on a mechanical system, bypassing the difficult problem of parameterizing the system degrees of freedom of reduced coordinates. They can handle constraints between the 1D elements and their environment (*absolute constraints*) or themselves (*relative constraints*). Baraff [Bar96] proposed a general and linear-time method to make them usable in an interactive context, but still quadratic for loop constraints. Within a dynamic framework, they are commonly solved with the Baumgarte stabilization scheme [Bau72], with the drawback of not imposing at once the exact realization of the constraint. Faure et al. [Fau99] proposed an iterative post-stabilization that re-apply constraints after the numerical integration phase. Qin *et al.* [QT96] and Lenoir *et al.* [LGM+04] applied Lagrange Multipliers to dynamic splines. Moreover, this model is adequate for specialized constraints like sliding constraints for suturing [LGMC05]. Lenoir [LF04] also proposed a method to achieve heterogeneous physical simulation of both deformable and rigid-bodies objects . However, the constrained system had to cope with a quadratic complexity with respect to the number of constraints. This implied a limited number of simultaneous real-time constraints.

Alternatively, physically-motivated penalty methods have also been proposed that compute constraint forces based on energy functions [WFB87, BB88], but they can result in stiff equations and local minima problems. The method presented in [GG94] manipulates positions instead of computing forces, but tended to yield non-physical effects [WT06]. Choe *et al.* [CCK05] modeled soft constraints by attaching a linear spring and an angular spring between each pair of links. They address the stiff problem of hair to the detriment of the mechanical accuracy. Without the need for complex remeshing in collision, plasticity and fracture, meshless or point-based methods have enjoyed recent popularity due to their flexibility. Their interesting formulation facilitates the handling of multiple and possibly conflicting constraints [MKN+04, MHHR07], but yet it is not clear in which measure such tools may be extended to provide constraints on models other than point-based.

Weinstein *et al.* [WT06] proposed a pre-stabilization method for articulated rigid bodies, allowing linear solving both in the number of bodies and in the number of auxiliary contact. They compute allowable trajectories before moving the rigid bodies to their new positions, instead of correcting them after the fact when it can be difficult to incorporate the effects of contact and collision. Gissler *et al.* [GBT06] improved its efficiency, dynamically enabling/disabling constraints on deformable mass-point systems. The proposed constraints may attach a point to a point, a line, and a surface. Their method employs information on the underlying numerical integration scheme, solving efficiently loops, but introduces external forces depending on the integration scheme. Goldenthal *et al.* [GHF+07] recently proposed a method that constrains extensibility, acting as a velocity filter, but not orientation.

Next section introduces the general framework we propose, as well as the method for combining Lagrangian constraints with displacement-based integration methods.

3 MICADO principle

This section describes the general framework of MICADO: we present the initial assumptions on the DODE model, and introduce the necessary notions to understand how to combine desired constraints with the displacement-based integration scheme.

3.1 Assumptions on the DODE model

As shown in section 2, many different models have been proposed for DODE simulation. Any DODE may be used with MICADO, provided its free mechanical system can be expressed with respect to the degree of freedom displacements. The linearized backward Euler scheme, that is classically used in computer graphics and virtual reality [BW98, DSB99], fulfills such a condition, providing a robust integration between two states n and $n+1$, separated by a time step Δt. Either dynamic or quasi-static system can be reduced to the simple following form :

$$\mathbb{K}\Delta^{n+1}\mathbf{X} = \mathbf{F} \tag{1}$$

where $\mathbb{K}$ is a stiffness matrix, $\mathbf{F}$ the system force vector and $\Delta^{n+1}\mathbf{X}$ the displacement vector between the current state and the following one. We make a second assumption on the DODE model: the position vector $\mathbf{x}$ must be expressed from the degrees of freedom $\mathbf{x}_i$ as follows: $\mathbf{x} = \sum_1^n b_i(u)\mathbf{x}_i$, with $\mathbf{x}_i$ the degrees of freedom, and $b_i(u)$ the corresponding blending functions, depending on the DODE model. In practice, this condition is not a problem: such b_i functions are defined in FEM and Dynamic Material Splines; we shall consider a simple Lagrange interpolation between the degrees of freedom in a mass-spring system.

3.2 Including constraints within a displacement-based solving scheme

A constraint may be formulated by a vector system $\mathbf{g}$, that depends on the position vector $\mathbf{x}$ and its related parametric coordinate u: $\mathbf{g}\left(\mathbf{x}(u), \mathbf{x}'(u), \mathbf{x}''(u), \dots\right) = 0$, where $\mathbf{x}'$ denotes the differentiation of the position with respect to the parametric coordinate u, and $\mathbf{x}'', \dots$, the further differentiations.

Using Lagrange multipliers, the corresponding constrained system is commonly stated as [Bar96]:

$$\begin{pmatrix} \mathbb{K} & \mathbb{J}^t \\ \mathbb{J} & \mathbb{O} \end{pmatrix} \begin{pmatrix} \Delta^{n+1}\mathbf{X} \\ -\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{F} \\ \mathbf{E} \end{pmatrix} \tag{2}$$

where $\mathbf{E}$ is the violation vector of the constraints, λ is the Lagrange multiplier vector, and $\mathbb{J}$ is the derivative of the constraint vector equations $\mathbf{g} = \mathbf{0}$ with respect to the coordinates, also called the Jacobian of the constraints: $\mathbb{J} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}_i}$. The violation vector $\mathbf{E}$ makes the imposed configuration evolve between the two succeeding states. A constraint remains unchanged between two states if its violation is set to null. The Lagrange multiplier λ corresponds to the required set of forces to keep the desired configuration between the two states.

In this rest of this paper, we propose a method to define and use efficiently a new constraint, while detailing Jacobian matrices $\mathbb{J}$ and violations $\mathbf{E}$. We have to perform three steps: first, establish its constraint equation; second, differentiate this equation with respect to the degrees of freedom $\mathbf{x}_i$ to identify the Jacobian matrix $\mathbb{J}$; third, determine the Lagrange multiplier λ signification, i.e. corresponding constraint force, and an efficient use of the violation E. After having considered Lagrangian constraints in general, we will address them specifically, depending on their nature: *absolute constraints* if they relate one DODE point to the environment, or *relative constraints* if they interrelate two or more DODE points.

4 Constraining a single point of a DODE

Absolute constraints relate one DODE point to the environment. Their size depends on the blending function b_i locality: for example, it yields 2 for most of the mass-spring and continuous models, 4 for Dynamic Material cubic B-Splines. In this section and in the rest of this paper, we use Einstein notation convention: anywhere the form $a_i b_i$ appears, it should be replaced without any ambiguity by $\sum_i a_i b_i$.

4.1 Manipulation Constraints:

to constrain all degrees of freedom of a DODE point $\mathbf{x}_0$ of u_0, we use both a position constraint $\mathbf{g}_\mathrm{p}$ (equation 3) and an orientation constraint $\mathbf{g}_\mathrm{o}$ (equation 4):

$$\mathbf{g}_\mathrm{p} = \mathbf{x}(u_0) - \mathbf{x}_0(u_0) \tag{3}$$

$$\mathbf{g}_\mathrm{o} = \mathbf{x}'(u_0) - \mathbf{x}'_0(u_0) \tag{4}$$

As the position and the orientation of the constrained DODE point may change, we differentiate $\mathbf{g}_\mathrm{p}$ and $\mathbf{g}_\mathrm{o}$ with respect to $\mathbf{x}_i$, according to Einstein notation

convention:

$$\mathbf{dg}_{\mathrm{p}} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}_i}\mathbf{dx}_i = \mathbb{I}_3 b_i(u_0)\mathbf{dx}_i \quad (5)$$

$$\mathbf{dg}_{\mathrm{o}} = \frac{\partial \mathbf{x}'}{\partial \mathbf{x}_i}\mathbf{dx}_i = \mathbb{I}_3 b_i'(u_0)\mathbf{dx}_i \quad (6)$$

where $\mathbb{I}_k$ denotes the identity matrix of size k. Numerically, the system yields:

$$\mathbf{E}_{\mathrm{p}} = \mathbb{J}_{\mathrm{p}}\Delta\mathbf{X} = \mathbb{I}_3 b_i(u_0)\Delta\mathbf{x}_i \quad (7)$$

$$\mathbf{E}_{\mathrm{o}} = \mathbb{J}_{\mathrm{o}}\Delta\mathbf{X} = \mathbb{I}_3 b_i'(u_0)\Delta\mathbf{x}_i \quad (8)$$

Specifying the violations $\mathbf{E}_{\mathrm{p}}$ and $\mathbf{E}_{\mathrm{o}}$ provides easy and exact handling at the DODE point of u_0. The violation $\mathbf{E}_{\mathrm{p}}$ is a translation vector of the DODE point between two simulation states. The violation $\mathbf{E}_{\mathrm{o}}$ is the difference between the desired unit direction $\mathbf{t}^{n+1}$ and the current one $\mathbf{t}^n$, multiplied by $\|\mathbf{x}'(u_0)\|$ to prevent the violation from introducing local stretching:

$$\mathbf{E}_{\mathrm{o}} = \|\mathbf{x}'(u_0)\|\left(\mathbf{t}^{n+1} - \mathbf{t}^n\right) \quad (9)$$

The sum of the Lagrange multipliers λ_{p} and λ_{o} correspond to the required force to impose the specified position and orientation.

4.2 Subspace constraints

Subspace constraints let a DODE point move freely in a subspace like a plane or a sphere. Their equation is usually composed of one or more scalar products. In fact, any kind of equation may be used, provided that it is defined for the considered DODE point. Two or more of them can also be mixed to obtain a new subspace: for example, a circle is the intersection of plane and a sphere, whereas an axis is the intersection of two planes. Moreover, a trajectory may be modeled by succeeding linear approximations, such as axis constraints.

A *plane constraint* g_{p} is defined by a DODE point $\mathbf{x}(u_0)$ and a normal $\mathbf{n}$:

$$g_{\mathrm{pl}} = (\mathbf{x}(u_0) - \mathbf{x}_0(u_0)) \cdot \mathbf{n} = 0 \quad (10)$$

The differentiation of the plane constraint g_{pl} with respect to the degrees of freedom results in:

$$dg_{\mathrm{pl}} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}_i}\mathbf{dx}_i \cdot \mathbf{n} = b_i(u_0)\mathbf{dx}_i \cdot \mathbf{n} \quad (11)$$

Numerically, the plane constraint system yields:

$$E_{\mathrm{pl}} = \mathbb{J}_{\mathrm{pl}}\Delta\mathbf{X} = -b_i(u_0)\mathbf{n} \cdot \Delta\mathbf{x}_i \quad (12)$$

where $\mathbb{J}_{\mathrm{pl}}$ is a one row matrix. The violation E_{pl} corresponds to the magnitude of a translation of the DODE point along the normal $\mathbf{n}$, whereas the Lagrange

multiplier λ_{pl} is the magnitude of the force of direction $\mathbf{n}$, that imposes the plane constraint. To model an axis constraint, we use two plane constraints with non collinear normals.

A *sphere constraint* or *distance constraint* g_{d}, is defined by a space point $\mathbf{p}$ and the current DODE point $\mathbf{x}(u_0)$, of distance d:

$$g_{\mathrm{d}} = (\mathbf{x}(u_0) - \mathbf{p})^2 - d^2 = 0 \tag{13}$$

Differentiating the distance constraint equation g_{d} with respect to the degrees of freedom $\mathbf{x}_i$ results in:

$$dg_{\mathrm{d}} = 2\frac{\partial \mathbf{x}(u_0)}{\partial \mathbf{x}_i}(\mathbf{x}(u_0) - \mathbf{p}) \cdot \mathbf{dx}_i = 2b_i(u_0)\mathbb{I}_3(\mathbf{x}(u_0) - \mathbf{p}) \cdot \mathbf{dx}_i \tag{14}$$

Numerically, the distance constraint system yields:

$$E_{\mathrm{d}} = \mathbb{J}_{\mathrm{d}}\Delta\mathbf{X} = 2b_i(u_0)\mathbb{I}_3(\mathbf{x}(u_0) - \mathbf{p}) \cdot \Delta\mathbf{x}_i \tag{15}$$

The violation E_{d} increases or decreases the distance d between two succeeding states, whereas the Lagrange multiplier λ_{d} corresponds to the magnitude of the force in the direction of the sphere radius.

The previously defined constraints are absolute, i.e. with respect to the environment. We can also define them with respect to a DODE point, providing more or less intricate structures of one-dimensional elements.

5 Constraining several DODEs together

Relative constraints relate two or more DODE points to each other. Each kind of absolute constraint may be translated into a relative constraint.

5.1 Fusion constraints

Fusion constraints are the transposition of manipulation constraints. They involve two DODE points $\mathbf{x}_0$ and $\mathbf{x}_1$ of parametric coordinates u_0 and u_1, respectively. We compel the two DODE points to have the same space position, whereas the orientations of respective unit tangents are likely to be different. A fusion constraint is thus modeled by a relative position constraint $\mathbf{g}_{\mathrm{rp}}$ (equation 16) and an orientation constraint $\mathbf{g}_{\mathrm{ro}}$ (equation 17):

$$\mathbf{g}_{\mathrm{rp}} = \mathbf{x}(u_0) - \mathbf{x}(u_1) - \mathbf{x}_{10} \tag{16}$$

$$\mathbf{g}_{\mathrm{ro}} = \mathbf{x}'(u_0) - \mathbf{x}'(u_1) - \mathbf{x}'_{10} \tag{17}$$

Figure 3: Virtual prototyping: positioning of the two opposite extremities of a cable bundle. Its different parts are assembled with fusion constraints.

As the position and the orientation of the constrained DODE points may change, we differentiate $\mathbf{g}_{\text{rp}}$ and $\mathbf{g}_{\text{ro}}$ with respect to $\mathbf{x}_i$:

$$\mathbf{dg}_{\text{rp}} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}_i}\mathbf{dx}_i = \mathbb{I}_3\left(b_i(u_0) - b_i(u_1)\right)\mathbf{dx}_i \tag{18}$$

$$\mathbf{dg}_{\text{ro}} = \frac{\partial \mathbf{x}'}{\partial \mathbf{x}_i}\mathbf{dx}_i = \mathbb{I}_3\left(b'_i(u_0) - b'_i(u_1)\right)\mathbf{dx}_i \tag{19}$$

Numerically, the system yields:

$$\mathbf{E}_{\text{rp}} = \mathbb{J}_{\text{rp}}\Delta\mathbf{X} = \mathbb{I}_3\left(b_i(u_0) - b_i(u_1)\right)\Delta\mathbf{x}_i \tag{20}$$

$$\mathbf{E}_{\text{ro}} = \mathbb{J}_{\text{ro}}\Delta\mathbf{X} = \mathbb{I}_3\left(b'_i(u_0) - b'_i(u_1)\right)\Delta\mathbf{x}_i \tag{21}$$

Specifying the violation $\mathbf{E}_{\text{ro}}$ changes the rotation between the two tangent vectors $\mathbf{x}'(u_0)$ and $\mathbf{x}'(u_1)$, like absolute orientation constraints. Setting $\mathbf{E}_{\text{rp}}$ to a different other than null has no physical meaning. The sum of the Lagrange multipliers λ_{p} and λ_{o} correspond to the required force vector to hold the fusion constraint.

5.2 Other relative constraints

It is also possible to translate absolute subspace constraints into relative ones. For example, relative distance constraints preserve the distance between two DODE points and may be used as alternative rigid 1D elements (see figure 4); relative plane constraints can provide an interesting self-collision response. Until now, all constraints have been defined for a constant DODE point. We now propose sliding constraints that let the parametric coordinate vary with respect to the constrained DODE.

6 Sliding constraints on a DODE

We introduce here an additional variable in constraints, the DODE parametric coordinate. We consider it as a new degree of freedom. This allows us to model new kinds of constraints, called sliding constraints.

Figure 4: Deformation of a spidernet, modeled by axial DODEs and lateral distance constraints.

6.1 Keyhole constraint

Lenoir *et al.* [LGM+04] proposed smooth constraints, that impose a DODE to slide through a static space point, with or without friction. We called such a constraint a *keyhole constraint*, in reference to keyhole surgery. Its equation $\mathbf{g}_{\mathrm{k}}$ yields in our framework:

$$\mathbf{g}_{\mathrm{k}}(\mathbf{x}, u) = \mathbf{x}(u) - \mathbf{x}_0(u_0) = \mathbf{0} \tag{22}$$

Since both position and parametric coordinate can vary, we now differentiate this equation with respect to the degrees of freedom $\mathbf{x}_i$ and u:

$$\mathbf{g}_{\mathrm{k}}(\mathbf{x}, u) = \frac{\partial \mathbf{x}(u)}{\partial \mathbf{x}_i}\mathbf{d}\mathbf{x}_i + \frac{\partial \mathbf{x}(u)}{\partial u}du = \mathbb{I}_3 b_i(u)\mathbf{d}\mathbf{x}_i + \mathbf{x}'du \tag{23}$$

The keyhole constraint results numerically in:

$$\mathbf{E}_{\mathrm{k}}(\mathbf{x}, u) = \mathbb{J}_r \Delta\mathbf{x}_i + \mathbb{J}_u \Delta u = \mathbb{I}_3 b_i(u_0)\Delta\mathbf{x}_i + \mathbf{x}'\Delta u \tag{24}$$

The additional Lagrange multiplier Δu is thus associated to the keyhole constraint. It gives the parametric coordinate offset for constraint updating after system solving. The scalar violation E_u associated to Δu may impose a sliding between two states,whereas $\mathbb{J}_u$ may have a forth component c, corresponding to a friction that opposes to sliding. The violation $E_{\mathbf{k}}$ and the Lagrange multiplier $\lambda_{\mathbf{r}}$ associated to $\mathbf{x}$ coordinates have the same meaning as in a position constraint. Note that having the orientation slide is also possible, but has a more restrictive use. Nevertheless, keyhole constraints have their relative translation, that may be used to model a slipknot. Our formulation provides a smarter and more direct solving than Lenoir *et al.* [LGM+04] quadratic solving, which uses a Baumgarte scheme. Moreover, our time complexity is linear with the number of sliding constraints (see section 7.1).

To constraint also the orientation, we propose to add an orientation constraint. Since after each simulation step the orientation at the new parametric coordinate is not likely the same, we have to correct it with the initial tangent $\mathbf{x}'_{\mathrm{ref}}$. Combining equations 23 and 6 with the correction 9 yields:

$$\mathbf{d}\mathbf{g}_{\mathrm{cs}} = \mathbb{I}_3 b_i(u)\mathbf{d}\mathbf{x}_i + \mathbf{x}'_{\mathrm{ref}}du \tag{25}$$

$$\mathbf{d}\mathbf{g}_{\mathrm{co}} = \mathbb{I}_3 b'_i(u_0)\mathbf{d}\mathbf{x}_i - \mathbf{x}'_{ref} + \mathbf{x}'_0 \tag{26}$$

An update of the constraint parametric coordinate u_0 is still required after the simulation step. Using this formulation, we can constrain a DODE to slide through a specified point, with a specified orientation. We call such a constraint an *oriented keyhole*.

6.2 Pearl constraints

Another look at the equations involved in the keyhole constraint definition can be derived in a visually totally different behavior. The *pearl* constraints impose the motion of sliding mass points along the curve defined by the DODE, with or without friction. A first and naive approach is to differentiate the following constraint equation:

$$\mathbf{g}_{\mathrm{sp}}(\mathbf{x}, u) = \mathbf{x}_0(u) - \mathbf{x}_0(u_0) = 0 \tag{27}$$

This results in:

$$\mathbf{dg} = \mathbf{x}' du \tag{28}$$

Please note that contrary to the keyhole constraint, the position vector $\mathbf{x}$ does not depend on the degrees of freedom of the DODE. However, the vector $\mathbf{x}'$ may have null coordinates, causing one or two null equations and thus an indefinite system. Moreover, this is an oversized and overcost solution of a 1D problem. Such a problem should be solved with one equation. So we propose to project locally the 3D constraint equations onto the DODE. We use the forward dynamic Euler scheme of a particle [DSB99]:

$$\left(\frac{c}{\Delta t} + \frac{m}{\Delta t^2}\right) \Delta^{n+1}\mathbf{x_p} = \mathbf{F}_{sp} + \frac{m}{\Delta t}{\mathbf{v_p}}^n \tag{29}$$

with the particle position x_p, its velocity v_p and its mass m, and $\mathbf{F}_{sp}$ the sum of external forces applied on the sliding mass. The projection of this equation onto the DODE results in:

$$\left(\frac{c}{\Delta t} + \frac{m}{\Delta t^2}\right) \Delta^{n+1} u = \mathbf{F}_{sp} \cdot \frac{\mathbf{x}'}{\|\mathbf{x}'\|} + m\frac{\Delta^n u}{\Delta t^2} \tag{30}$$

This weak coupling compels us to add an external force $\mathbf{f}_{sp}$ to the material DODE, corresponding to $\mathbf{F}_{sp}$ and the friction c between the sliding mass and the DODE:

$$\mathbf{f}_c = \mathbf{F}_{sp} - c\frac{\Delta^n u}{\Delta t}\frac{\mathbf{x}'}{\|\mathbf{x}'\|} \tag{31}$$

7 Applications and results

We describe here practical situations where the proposed tools can be useful. We first discuss the "right" method for proper balancing of the numerical system. We then describe all the provided illustrations, along with additional explanations, when necessary.

7.1 Efficient numerical solving

Whether one, or several DODEs are involved in the simulation is not of matter for numerical efficiency. All the degrees of freedoms are classically put all together in equation 2. It is known [Bar96] that equation 2 is not the optimal ordering for numerical solving.

For efficient solving, the classical approach is to reduce the bandwidth of the numerical system using the reverse Cuthill-McKee algorithm, classical in the applied mathematics community [Bar96, CM69]. This algorithm works as follows: first, a relation graph is constructed from the linear system: each node of the graph relates to an unknown; in that graph, an edge exist between node i with node j iff $\mathbb{K}(i,j) \neq 0$. In this graph, a *peripheral* node is selected (i.e. a node whose distance to some other node in the graph equals the maximum distance between any two nodes in the graph). From that node, a breadth first search is achieved; the new unknown numbering is directly provided by this search (reversing the produced list gives better results in practice).

It is known that free DODE systems result in banded matrices: efficiency, linear-time solving are easily available for such systems, e.g. using LU Band decomposition. When using MICADO, reordering is required because of relative constraints: we choose to apply the Cuthill-McKee algorithm on the graph that is generated, considering that constraint equation only involves one degree of freedom per DODE point involved: rigorously speaking, this is wrong since, as mentioned in section 3.1, each DODE sample can involve several degrees of freedom, depending on the interpolation scheme locality.

In practice, as shown on figure 5 and 6, using this simplified relation graph provides good results.

When constraints are dynamic (e.g. constraints that move on the DODE such as sliding one, or constraints that are dynamically released, e.g. bow and arrow), global system assembly and reordering is needed at each time-step. Actual reordering does not imply any actual matrix re-composition: a simple vector is used as a look-up table for this reordering. Constraints described in sections 4 are very easily inserted in the system (on the middle of corresponding degrees of freedom), and do not actually need Cuthill-McKee algorithm. Constraints of section 5 (including loop creation) are handled using the described algorithm. Finally, all the proposed sliding point constraints, defined in section 6, shall be re-positioned throughout time for maintaining numerical efficiency: because of the look-up vector, this can be done in $O(1)$. The reordered system is then solved with a LU decomposition for banded matrix.

7.2 Examples

We have implemented the described constraints, and used them for a number of simulations. We base our examples on the DODEs described in [TGDM07].

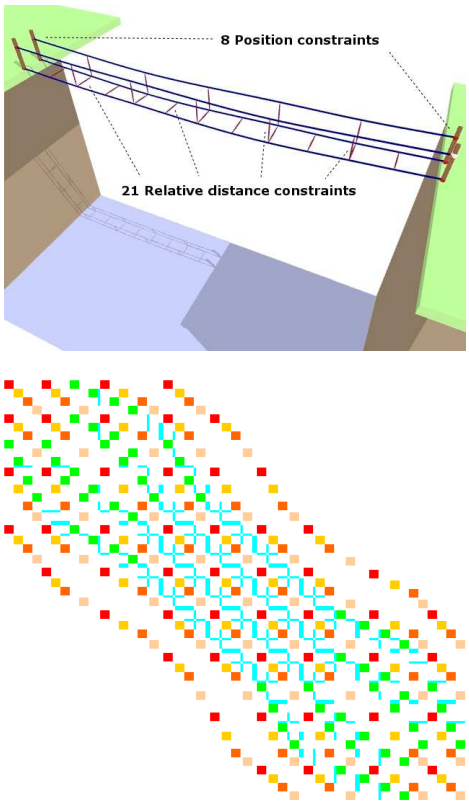

Figure 5: The bridge: used constraints and related banded matrix. Warm colors correspond to control points of the four ropes whereas cold colors yield the absolute constraints in green and the distance constraints between the rope in blue.

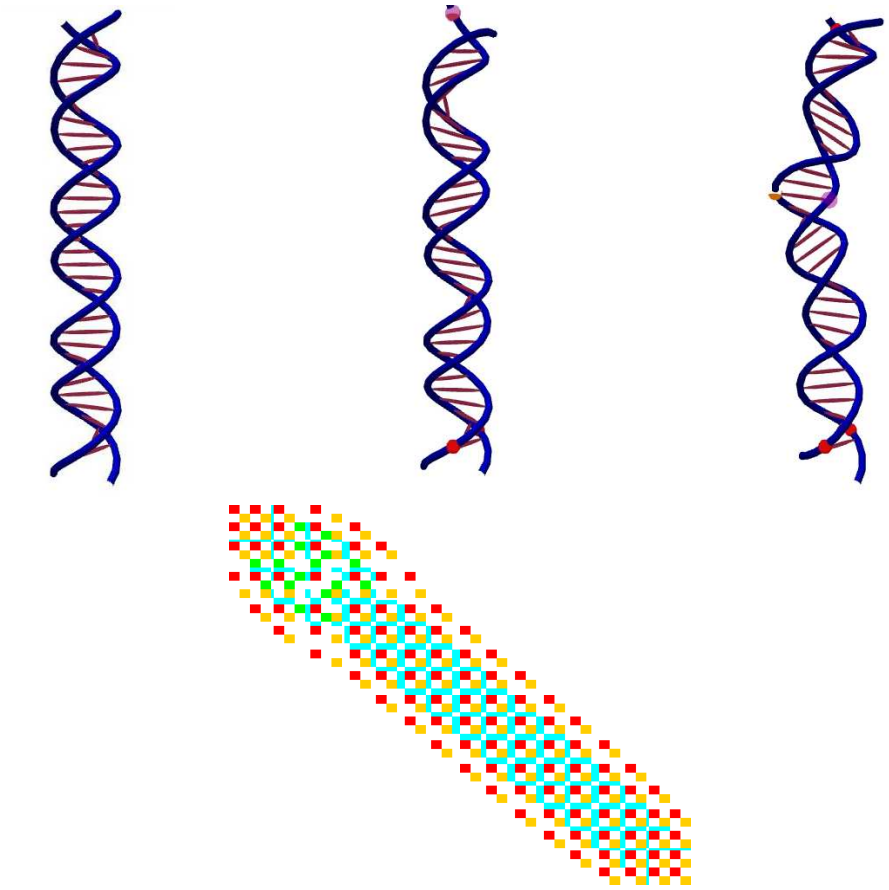

Figure 6: DNA modeling and the corresponding banded matrix. The two helices are DODEs, whereas Hydrogen bonds are modeled with relative distance constraints.

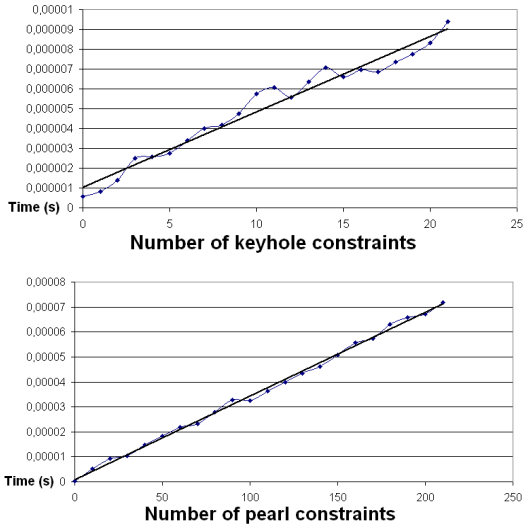

Figure 7: Linear-time constraint performances: computation times of keyhole and pearl constraints.

They were run on an Intel Centrino Duo laptop, with two 2Ghz processors and 1Gb of RAM. Constraint computation times are fast enough to provide relatively complex scenes at interactive rates 7.

The first application is related to virtual prototyping of electrical cable bundle positioning and clipping along the path of a lateral car structure. It involves manipulation constraints and fusion constraints (see figure 3). To our knowledge, this is the first modeling of such a mechanically valid handling, at interactive rates.

The second illustration is devoted to structures where "loops" are present: a rope bridge. In that structure, a large number of loops is involved. In the example shown, the ropes are linked (both for footpath and handrail linkage) using 21 distance constraints. Using such a constraint instead of a mechanical model provides a significant speed up, while maintaining realism (see figure 1). We provide the matrix structure of the bridge constrained at its extremities (see figure 5). Another example using distance constraints is the modeling of a spider net by axial DODEs and lateral distance constraints (see fig.4). Also in that case, interactive frame rate is reached. Pearl constraints provide a smart and efficient way to simulate objects sliding on a string. We illustrate their use with the threading of a pearl necklace (see figure 2). The use of such constraints also enables robust and efficient collision handling, since their equation of motion is restricted to the DODE subspace. The detection between pearls only involves a list of parametric coordinates, whereas the response is easily modeled with impulse forces for elastic shocks and simple Lagrangian constraints for contacts.

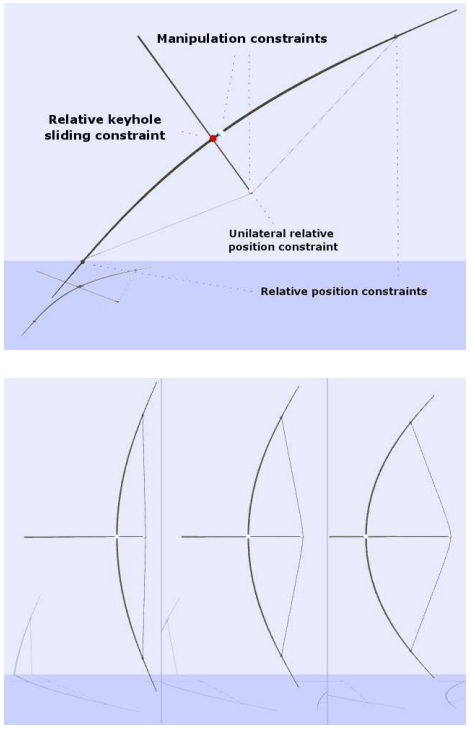

Figure 8: Bow simulation: modeling by constraints and extension.

The final example illustrates a practical example that is classically not simulated using complete deformable sets: the mechanical system is composed of an arrow, and a bow. We reproduce the behavior of a bow that projects an arrow (see figure 8), using three DODEs for the limbs, the string and the arrow. The limbs are connected to the string using two relative constraints. The arrow is linked to the limbs using a relative keyhole constraint. An unilateral relative constraint is used to model the contact between the bow string, and the arrow is released. Elastic energy is stored within the limbs of the bow and transformed into motion when the string is released. That way, the string transfers its internal force to the arrow. The global behavior is visually realistic, including the vibrations of the string shortly after the arrow release, as well as the arrow trajectory that is influenced by the bow orientation.

All these animations run at interactive rates, with an average cost of about $1ms$ per simulation step. Using unknown reordering described in the previous subsection, efficient solving is reached (see table 1).

Experiment	Absolute λ	Relative λ	cost
Bridge	32	21	1.7ms
Necklace	8	10	0.4ms
Bow	11	16	0.56ms
Cable bundle	7	35	1ms
Spider net	28	49	4.0ms
DNA	8	32	1.1ms

Table 1: Practical computation cost of the proposed examples with our method.

8 Conclusion

In this paper we proposed a set of useful Lagrangian constraints, that can be potentially used with most available one-dimensional deformable models. They are expressed based on displacement, which makes them particularly efficient for numerical solving. The proposed tools allow for real-time complex structure simulation (such as a bridge or a bow), and also novel interactions between simulated objects (e.g. pearl threading). The set of possible useful constraints is far from being completely defined; as a matter of fact we think this first proposal can open the way to generalized use of 1D-deformable models. The use of Lagrangian constraint as a tool for dynamic LOD within interactive scenes is especially interesting to us, and we intend to pursue this as a future work. We could also think of extending this work within the context of skeleton based animation.

Acknowledgements

References

[AUK92] Ken-ichi Anjyo, Yoshiaki Usami, and Tsuneya Kurihara. A simple method for extracting the natural beauty of hair. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 111–120, New York, NY, USA, 1992. ACM Press.

[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. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 1180–1187, New York, NY, USA, 2006. ACM Press.

[Bar96] David Baraff. Linear-time dynamics using lagrange multipliers. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on*

Computer graphics and interactive techniques, pages 137–146, New York, NY, USA, 1996. ACM Press.

[Bau72] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Comp. Meth. in Appl. Mech. and Eng.*, 1:1–16, 1972.

[BB88] Ronen Barzel and Alan H. Barr. A modeling system based on dynamic systems. *Computer Graphics (SIGGRAPH'88 Proceedings)*, 22:179–188, 1988.

[BP05] F. Boyer and D. Primault. Finite element of nonlinear cables : applications to robotics. *Far East Journal of Applied Mathematics*, 19(1):1–34, 2005.

[BW98] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54, New York, NY, USA, 1998. ACM Press.

[CCK05] Byoungwon Choe, Min Gyu Choi, and Hyeong-Seok Ko. Simulating complex hair with robust collision handling. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 153–160, New York, NY, USA, 2005. ACM Press.

[CDL+05] Stphane Cotin, Christian Duriez, Julien Lenoir, Paul Neumann, and Steven Dawson. New approaches to catheter navigation for interventional radiology simulation. In *Proceedings of Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 534–542, Palm Springs, California, USA, 26-30 october 2005.

[CM69] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th national conference*, pages 157–172, New York, NY, USA, 1969. ACM.

[DSB99] Mathieu Desbrun, Peter Schröder, and Alan Barr. Interactive animation of structured deformable objects. In *Proceedings of Graphics interface*, pages 1–8, 1999.

[Fau99] François Faure. Interactive solid animation using linearized displacement constraints. In B. Arnaldi and G. Hégron, editors, *Computer Animation and Simulation'98*, pages 61–72, 1999.

[Fea87] Roy Featherstone. *Robot Dynamics Algorithms*. Kluwer Academic Publishers, 1987.

[GBT06] M. Gissler, M. Becker, and M. Teschner. Local constraint methods for deformable objects. In *Proc. Virtual Reality Interactions and Physical Simulations VriPhys*, pages 25–32, Madrid, Spain, Nov. 6-7 2006.

[GG94] Jean-Dominique Gascuel and Marie-Paule Gascuel. Displacement constraints for interactive modeling and animation of articulated structures. *The Visual Computer*, 10(4):191–204, 1994.

[GHF+07] Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. Efficient simulation of inextensible cloth. *ACM Trans. Graph.*, 26(3):49, 2007.

[GLM06] Russell Gayle, Ming C. Lin, and Dinesh Manocha. Adaptive dynamics with efficient contact handling for articulated robots. In *Robotics: Science and Systems*, 2006.

[GS06] Mireille Grégoire and Elmar Schömer. Interactive simulation of one-dimensional flexible parts. In *SPM '06: Proceedings of the 2006 ACM symposium on Solid and physical modeling*, pages 95–103, New York, NY, USA, 2006. ACM Press.

[GS07] Mireille Grégoire and Elmar Schömer. Interactive simulation of one-dimensional flexible parts. *Comput. Aided Des.*, 39(8):694–707, 2007.

[Had06] Sunil Hadap. Oriented strands: dynamics of stiff multi-body system. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 91–100, 2006.

[LF04] Julien Lenoir and Sylvre Fonteneau. Mixing deformable and rigid-body mechanics simulation. In *Computer Graphics International*, pages 327–334, Hersonissos, Crete - Greece, June 16-19 2004.

[LGM+04] Julien Lenoir, Laurent Grisoni, Philippe Meseure, Yannick Rémion, and Christophe Chaillou. Smooth constraints for spline variational modeling. In *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 58–64, New York, NY, USA, 2004. ACM Press.

[LGMC05] Julien Lenoir, Laurent Grisoni, Philippe Meseure, and Christophe Chaillou. Adaptive resolution of 1d mechanical b-spline. In *Proceedings of Graphite*, pages 395–403, Dunedin - New Zealand, 29 november-2 december 2005.

[LMGC02] Julien Lenoir, Philippe Meseure, Laurent Grisoni, and Christophe Chaillou. Surgical thread simulation. In Marc THIRIET, editor, *Proceedings of Modelling and Simulation for Computer-aided Medicine and Surgery (MS4CMS)*, pages 102–107, Rocquencourt (France), 12-15 nov 2002. EDP Sciences.

[MHHR07] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *J. Vis. Comun. Image Represent.*, 18(2):109–118, 2007.

[Mil88] Gavin S. P. Miller. The motion dynamics of snakes and worms. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 169–173, New York, NY, USA, 1988. ACM.

[MKN+04] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. Point based animation of elastic, plastic and melting objects. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 141–151, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[NR01] O. Nocent and Y. Remion. Continuous deformation energy for dynamic material splines subject to finite displacements. In *CAS '01: Proceedings of the Eurographic workshop on Computer animation and simulation*, pages 88–97, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[Pai02] Dinesh K. Pai. Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum, Proceedings of Eurographics'02.*, 21:347–352, 2002.

[PLK02] J. Phillips, A. Ladd, and L.E. Kavraki. Simulated knot tying. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 1, pages 841–846, 2002.

[QT96] Hong Qin and D. Terzopoulos. D-nurbs: a physics-based framework for geometric design. *Visualization and Computer Graphics, IEEE Transactions on*, 2:85–96, 1996.

[RGL05] Stephane Redon, Nico Galoppo, and Ming C. Lin. Adaptive dynamics of articulated bodies. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 936–945, New York, NY, USA, 2005. ACM Press.

[RNG99] Yannick Rémion, Jean-Michel Nourrit, and Didier Gillard. Dynamic animation of spline like objects. In V. Skala, editor, *WSCG'99 Conference Proceedings*, 1999.

[RNN01] Y. Remion, J.-M. Nourrit, and O. Nocent. D-dimensional parametric models for dynamic animation of deformable objects. *The Visual Computer Journal*, 17(3):167–178, oct 2001.

[ST07] J. Spillmann and M. Teschner. Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 63–72, 2007.

[SVQ88] J. C. Simo and L. Vu-Quoc. On the dynamics in space of rods undergoing large motions: a geometrically exact approach. *Computer Methods in Applied Mechanics and Engineering*, 66:125–161, 1988.

[TGDM07] Adrien Theetten, Laurent Grisoni, Christian Duriez, and Xavier Merlhiot. Quasi-dynamic splines. In Bruno Lvy and Dinesh Manocha, editors, *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 409–414, New York, NY, USA, 2007. ACM Press.

[TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214, New York, NY, USA, 1987. ACM Press.

[WBK+07] Kelly Ward, Florence Bertails, Tae-Yong Kim, Stephen R. Marschner, and Marie-Paule Cani. A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):213–234, 2007. Member-Ming C. Lin.

[WFB87] Andrew Witkin, Kurt Fleischer, and Alan Barr. Energy constraints on parameterized models. *SIGGRAPH Comput. Graph.*, 21(4):225–232, 1987.

[WH04] Hidefumi Wakamatsu and Shinichi Hirai. Static modeling of linear object deformation based on differential geometry. *The International Journal of Robotics Research*, 23:293–311, 2004.

[WT06] Rachel Weinstein and Joseph Teran. Dynamic simulation of articulated rigid bodies with contact and collision. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):365–374, 2006. Member-Ron Fedkiw.

[WW90] Andrew Witkin and William Welch. Fast animation and control of nonrigid structures. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 243–252, New York, NY, USA, 1990. ACM Press.

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399