Smooth constraints for spline variational modeling

Julien Lenoir* IRCICA-LIFL, Lille
Laurent Grisoni† IRCICA-LIFL, Lille
Philippe Meseure‡ SIC, Poitiers
Yannick Rémion§ LERI, Reims
Christophe Chaillou¶
ALCOVE Project, INRIA Futurs, IRCICA-LIFL,
University of Lille 1, FRANCE

Abstract

This article introduces a new class of constraints for spline variational modeling, which allows more flexible user specification, as a constrained point can "slide" along a spline curve. Such constraints can, for example, be used to preserve correct parameterization of the spline curve. The spline surface case is also studied. Efficient numerical schemes are discussed for real-time solving, as well as interactive visualization during the energy minimization process. Examples are shown, and numerical results discussed.

CR Categories: I.3.5 [COMPUTER GRAPHICS]: Computational Geometry and Object Modeling—Physically based modeling; I.3.5 [COMPUTER GRAPHICS]: Computational Geometry and Object Modeling—Splines; J.6 [COMPUTER-AIDED ENGINEERING]: Computer-aided design; I.6.5 [SIMULATION AND MODELING]: Model Development—Modeling methodologies; I.6.0 [SIMULATION AND MODELING]: Model Development—General

Keywords: variational modeling, smooth constraint, spline, physically based modeling, dynamic resolution

1 Introduction

Among the important properties often appreciated in geometric modeling, stands the ability for a given model to be easily edited. This property partially explains the popularity of models such as splines [Piegl and Tiller 1997] and subdivision surfaces [Zorin and Schroder 2000]. Yet, in some special cases, many degrees of freedom for a geometric model can be quite a drawback to design simple, natural shapes, for example. Such a manipulation can be quite long and difficult for the user, making the model not suitable for such a goal. In order to solve this problem, deformation tools [Coquillart 1990; Barr 1984] are often used in order to provide simple edition for models with many degrees of freedom. Some models also have multiresolution features [Forsey and Bartels 1988; Grisoni et al. 1999] that provide a control of the number of degrees of freedom used during manipulation. This way, the user only manipulates the model at the desired resolution, and can precisely control the influence of the modifications.

Another possible solution is to use variational modeling [Welch and Witkin 1992; Gortler and Cohen 1995; Witkin et al. 1987], which principle is to combine initial condition to be respected, and a physical solver that determines the shape fitting the conditions and minimizing some energy criterion (classically, the global curvature). Constraints can thus be defined dynamically on the model to help the designer during the manipulation. The most popular constraints are those which affect a specific point of a curve or a surface (for instance, a fixed point, tangent, normal, etc...). Other constraints show themselves useful for direct manipulation like constraining a curve or surface to pass through a specific point (in free space or on an object). This has been already proposed in the past for particular objects.

It can be useful to easily define some sliding points on a shape, i.e. ensure that some point of the shape is at a specified location in space, whatever the corresponding parameter value of the point. For example, some applications in direct edition process, by fixing enough sliding points on the curve, we can entirely define its shape by specifying a kind of path. Such a constraint ensures that parameterization will also be taken into account during the energy minimization step. Apart from the parameterization point discussed above, such a constraint would offer additional degrees of freedom to variational modeling. This article presents a method for handling such constraints.

We use Lagrange formalism to describe our technique, and introduce what we call *smooth constraints* [1]. Classically in variational modeling, a static resolution process is used during the energy minimization step. This ensures a proper system resolution, but presents the drawback that the user does not have any control during the resolution process. We propose another approach, that uses dynamic resolution, and allows the user both to interact with the model during energy minimization, and control the numerical evolution of the system. In order to allow an efficient solving, we propose a modification of the constraint equations.

The paper is organized as follows: Section 2 consists in an overview of the previous work followed by section 3 presenting the principle of variational modeling, including notations and general equations. The spline case is especially developed in a sub-section. The next two sections present the theory for smooth spline curve constraints and the equations modifications needed to achieve high performance real-time resolution. A set of smooth constraints for a spline curve is then exposed in a section 6. Section 7 generalizes the theory to spline surfaces. Finally, numerical results, and examples, are presented in section 8 before concluding.

*email: Julien.Lenoir@lifl.fr
†email: Laurent.Grisoni@lifl.fr
‡email: philippe.meseure@sic.sp2mi.univ-poitiers.fr
§email: yannick.remion@univ-reims.fr
¶email: Christophe.Chaillou@lifl.fr

[1] It is to note that the term 'smooth constraint' is used in [Robinson 2003], associated to a different meaning. In our context, 'smooth constraint' refers to constraints specified without explicit parameter value (e.g. spline curve that slides through static space point).

2 Previous Work

In a general way, variational modeling searches the ideal emplacement of a model by minimizing an energy criterion (for example, the thin plate energy employed by Pottmann et al. [2002] for an active contour application). This approach can be done statically [Witkin et al. 1987] or dynamically [Qin and Terzopoulos 1996]. The employed energy can be seen in term of constraints that can be solved by a physical simulation. The interest in variational modeling resides in the possible constraints that can be applied to a model to ease its manipulation by the end-user. But creating constraints dynamically can yield some problems like discontinuity or an unsolvable system due to the resolution process. In this perspective, Gleicher [1992] proposed among others a direct edition method for constraints definition.

Welch and Witkin [1992] characterize a surface with a spline formulation and define an energy criterion based on a simplified expression from Terzopoulos et al. [1987]. Then, they define geometric constraints, finite-dimensional constraints and also transfinite constraints dealing with the entire surface (global energy). The resolution process is static and such proposed constraints do not allow a specific point of one object to be locked on another object surface (global consideration of the surface with evolving localization).

Among convenient constraints, Witkin et al. [1987] proposed higher levels tools. One of them obliges an object surface to pass through a specific point without specifying the actual object point. Their object can be defined by different ways, set of discrete points, iso-surface of implicit functions. By this way, a specific point of an object can easily be locked on the surface of another object defined by its implicit form. This is a pioneering work and a fundamental solution to the smooth constraint problem. The solution proposed a very interesting form of constraint but it has the drawback that the object defined by its implicit form is known by its surface. Nevertheless, it is in no way a trivial purpose to define a curve by an implicit form because it is not a natural way to create a 1D model especially for a deformable model. Even the use of a distance map brings computation time issues. Moreover, the static process of resolution needs energy local minima and thus the object configuration can jump from a solution to another one discontinuously.

Qin and Terzopoulors [1996] proposed a dynamic NURBS to simulate a curve and adapted it for geometric modeling by exploiting force reaction to bring local deformation and gravity to help designers. By a tensor product, they extend their tool to surface. They enforce their constraint by the way of Lagrange multipliers and Baumgarte stabilization scheme. They use augmented Lagrange technique but constraints which are not localized on the model can not be simulated by such method combination. The dynamic formalism proposed by Qin is interesting because it offers a temporal coherence of the behavior and thus avoid jumps which occur in a static process. We propose to complete this dynamic formalism in order to take into account a new class of constraints.

3 Dynamic Variational Modeling

3.1 Generalities

Traditionally, variational modeling consists in considering an objective function to be minimized, along with some specific constraints [Welch and Witkin 1992]. The objective function is generally based on an energy expression of the curve, surface or volume.

All these considerations can be simulated thanks to the lagrangian theory, that aims at minimizing the energy of a system in a dynamic process. Moreover, it can be directly extended to take into account some specific constraints, as in the classic way of variational modeling, by use of the Lagrange multipliers. In order to animate a curve, we consider this object as a set of degrees of freedom introduced in a mechanical system. With these degrees of freedom, we explain the mechanical law thanks to the lagrangian equations [Lenoir et al. 2002; Rémion et al. 1999]:

$$\begin{cases} \forall i, \frac{d\frac{\partial K}{\partial \dot{q}_i}}{dt} + \frac{\partial K}{\partial q_i} = Q_i + L^T.\lambda \\ \phi(q_i, \dot{q}_i) = 0 \end{cases} \quad (1)$$

where K is the kinetic energy of the system, q_i the degrees of freedom, Q_i the power of the different potential forces. L is a matrix defined using the different constraints (ϕ) relatively to all degrees of freedom [Rémion 2000]. λ are the Lagrange multipliers, each of them is related to the force intensity needed to preserve an associated constraint.

Classically, equations (1) are derived into a linear system:

$$\begin{pmatrix} M_g & L^T \\ L & 0 \end{pmatrix} \begin{pmatrix} \mathbf{A} \\ -\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{B} \\ \mathbf{E} \end{pmatrix} \quad (2)$$

where M_g is the generalized mass matrix of the system ones considers (resulting from the kinetic energy term), L the constraints matrix, A the acceleration of the degrees of freedom, B a vector that sums the different contributions of all external and inertial forces and finally E a vector coding the intensity of the violation of the different constraints.

By this way, we can fix a constraint directly on the degrees of freedom or anywhere on the object (degrees of freedom are not necessarily on the object as in the BSpline case) or between different objects present on the same dynamic system (like two patches with a common edge).

3.2 Spline Variational Modeling

The chosen model is a spline geometry on which we apply the mechanical laws. For a 1D spline, the position of a point on the curve is defined as a function of s (its parametric abscissa):

$$P(s,t) = \sum_{i=1}^{n} q_i(t).b_i(s) \quad (3)$$

where q_i are the control points, n the number of control points, b_i the basis functions linked to the spline type, s the parametric abscissa of the point and t the time (for our test, we used Catmull-Rom spline, cubic uniform BSpline and non uniform BSpline).

To simulate a spline, we define the three coordinates q_i^α of each of its control points as the degrees of freedom[2] of the dynamic system which thus has a size of $3n$. After some computations, we obtain:

$$\begin{aligned} \forall i, \frac{d\frac{\partial K}{\partial \dot{q}_i^\alpha}}{dt} &= m.\sum_{j=1}^{n}\int_0^1 b_i(s).b_j(s).ds.\ddot{q}_j^\alpha \quad (4) \\ \forall i, \frac{\partial K}{\partial q_i^\alpha} &= 0 \end{aligned}$$

where m is the mass of the object. We thus obtain a diagonal block generalized mass matrix with the same block for each axis:

$$M_g = \begin{pmatrix} M & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & M \end{pmatrix} \quad (5)$$

[2] thus, their accelerations are noted $\ddot{q}_i^\alpha$, which is equivalent to $\frac{d^2 q_i^\alpha}{dt^2}$

thanks to the equations (4), we easily define the components of M by:

$$M_{ij} = m.\int_0^1 b_i(s).b_j(s).ds \tag{6}$$

What we desire here is both the global and local modification of the curve when the end user changes the configuration. We can obtain a local modification by a direct edition of a point but the global shape of the curve does not change. We need here an internal energy permitting the propagation of the local modification to the entire curve. Since, we are interested in doing variational modeling, gravity is not involved. Q_i represents the influence of some springs that are associated to the curve for the internal energy [Lenoir et al. 2002], and the forces due to the interaction with the user. Those equations can also add forces given by a scene collision detection or self-collision process when needed.

The constraints are taken into account via the Lagrange multipliers λ. They allow to constrain a system, by the way of its degrees of freedom. We can thus constrain directly the degrees of freedom or any point of the object (for a BSpline, the degrees of freedom are the control points which do not belong to the curve). For example we can fix coordinates of any point on the curve by applying:

$$FixedPointConstraint(s_0,t) = P(s_0,t) - A$$

which declines in three simpler constraints, one for each axis. By the same way, constraints for fixing one or two coordinates are possible.

According to existing literature, such constraints can only be relative to a specific point of the curve (or more basically on the degrees of freedom themselves). We are thus not able to consider a constraint onto the entire curve. With such constraints, it is not possible to constrain the curve to pass through a point in space without specifying which constant point of the curve should be placed there. The next section presents a solution to this problem.

4 Smooth spline constraints

We wish to give more flexibility to the constraints by making them non localized on the curve. So that the effective point where the constraint is applied depends on the dynamics itself.

One example, is the sliding point constraint, where some point of the object must be at a specific location, but this curve point is not always the same step after step, depending on the dynamics. In this example, we just have to fix a point of the curve, considering the fact that it is not always the same point that is fixed during the process stabilization. As a curve point is defined by a parametric abscissa s, the specific s parameter of the fixed point constraint must be dynamic and thus depends on time : $s(t)$.

Clearly, our constraint is a fixed point constraint based on a dynamic abscissa parameter. Such a constraint g is written:

$$g(q,\dot{q},t,s(t)) = P(s(t),t) - P_0 \tag{7}$$

This equation imposes that some point of the spline must be at the position P_0. But it is a dynamic system so, the $s(t)$ will change over time in order to find the right point of the curve that minimizes the energy of the constrained system.

A problem occurs: The g equation demands to the system to verify a condition but the system ensures that the constraint will be fulfilled at a stable state but not necessarily immediately. This introduces a numerical drift on the constraint over time. To take into account this feature , we use the equation of g to formulate a second order differential equation that gives us a damp solution with a critical damping [Rémion et al. 1999; Baumgarte 1972]. This leads to the constraint equation:

$$\ddot{g} + \frac{2}{\delta t}\dot{g} + \frac{1}{\delta t^2}g = 0 \tag{8}$$

where δt is the time step used during the simulation.

By considering the equation (7), we obtain the suitable constraint equation:

$$\frac{d^2P}{dt^2} + \frac{2}{\delta t}\frac{dP}{dt} + \frac{1}{\delta t^2}(P - P_0) = 0 \tag{9}$$

In the development of this equation (using expression of P given by equation (3), new terms appear that do not exist for a simple fixed point constraint:

$$\begin{aligned}\sum_{i=1}^{n}(\ddot{q}_i(t).b_i(s(t)) + q_i(t).b_i'(s(t)).\ddot{s}(t)) = \\ - \sum_{i=1}^{n}(2.\dot{q}_i(t).b_i'(s(t)).\dot{s}(t) + q_i(t).b_i''(s(t)).\dot{s}(t)^2) \\ - \frac{2}{\delta t}\sum_{i=1}^{n}(\dot{q}_i(t).b_i(s(t)) + q_i(t).b_i'(s(t)).\dot{s}(t)) \\ - \frac{1}{\delta t^2}(\sum_{i=1}^{n} q_i(t).b_i(s(t)) - P_0)\end{aligned} \tag{10}$$

In fact, we can see that the constraint equation needs the dynamics of the s parameter. We thus have to consider it as an unknown of our system. It will be numerically integrated step by step like all other unknowns in order to determine its new position and velocity. This is a particular use of the lagrangian formulation because we just define a new unknown that is neither a degree of freedom nor a Lagrange multiplier. Because of this new unknown we have to find an additional equation in order to entirely define the system equation. This equation can give us an idea of the evolution of the s parameter or explain a constraint, linking s to the physical system.

If we consider a perfect constraint without friction, the lagrangian theory imposes that the virtual power of the strain due to this link must be equal to zero. In other words, the force generated by the Lagrange multipliers must not work in mechanical terms. For that, the Lagrange multipliers λ of this constraint must verify the equation:

$$\lambda.\frac{\partial g}{\partial s} = 0 \tag{11}$$

This theoretical framework is explained in more details in [Rémion 2003].

We obtain a global system:

$$\begin{pmatrix} M & 0 & 0 & 0 & Lx^T \\ 0 & M & 0 & 0 & Ly^T \\ 0 & 0 & M & 0 & Lz^T \\ 0 & 0 & 0 & 0 & Ls^T \\ Lx & Ly & Lz & Ls & 0 \end{pmatrix} \begin{pmatrix} \mathbf{A^x} \\ \mathbf{A^y} \\ \mathbf{A^z} \\ \ddot{\mathbf{s}} \\ -\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{B^x} \\ \mathbf{B^y} \\ \mathbf{B^z} \\ \mathbf{0} \\ \mathbf{E} \end{pmatrix} \tag{12}$$

the new matrix Ls describes the different constraints according to the new unknowns $\ddot{\mathbf{s}}$. In other words, it introduces the sliding point constraint into the system. Ls is composed by the terms of the constraint equations (10) where $\ddot{s}$ appears. For a single sliding point constraint, Ls is a (3×1) matrix composed by the column vector $\frac{\partial g}{\partial s}$.

5 Resolution

The equation system (12) is quite complex to solve because we do not have a direct relation that gives the new value of $\ddot{s}$. For an interactive application, such a resolution can degrade the performance and thus make the manipulation difficult because of the latency of the animation. Consequently, we choose to consider equation (11) in a different way.

We accept that the effective work of the force generated by the Lagrange multipliers is not null and we represent it as an error. This error is due to an incorrect value of s and is applied to correct the dynamics of this parameter. This approach gives us an equation slightly different from equation (11):

$$\varepsilon.\ddot{s} - \lambda.\frac{\partial g}{\partial s} = 0 \tag{13}$$

In order to stay close to the theoretical framework, we multiply the acceleration of s by a factor ε close to zero. This little modification of the equation gives this new system of equations:

$$\begin{pmatrix} M & 0 & 0 & 0 & Lx^T \\ 0 & M & 0 & 0 & Ly^T \\ 0 & 0 & M & 0 & Lz^T \\ 0 & 0 & 0 & \varepsilon & Ls^T \\ Lx & Ly & Lz & Ls & 0 \end{pmatrix} \begin{pmatrix} \mathbf{A^x} \\ \mathbf{A^y} \\ \mathbf{A^z} \\ \ddot{\mathbf{s}} \\ -\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{B^x} \\ \mathbf{B^y} \\ \mathbf{B^z} \\ \mathbf{0} \\ \mathbf{E} \end{pmatrix}$$

In this system, the matrix L is defined by its components: $(LxLyLz)$. Ls is an extension of L which permits to explain constraints on the $\ddot{s}$ unknowns.

The equation (13) gives us a direct relation to find the value of the $\ddot{s}$ unknown and thus accelerates the resolution process.

We decide to solve the system by decomposing the acceleration in two parts, one of *tendency* and one of *correction*: $A = A_t + A_c$ [Rémion 2000]. The acceleration of tendency represents the acceleration without any constraint and the other acceleration is the correction due to the constraints. This leads us to this new equation system:

$$\begin{cases} M.A_t^x = B^x \\ M.A_t^y = B^y \\ M.A_t^z = B^z \\ M_g.A_c = L^T.\lambda \\ \varepsilon.\ddot{s} = Ls^T.\lambda \\ L.(A_t + A_c) + Ls.\ddot{s} = E \end{cases}$$

We replace the terms A_c and $\ddot{s}$ in the sixth equation by their expression respectively in the fourth equation and the fifth equation. These replacements yield an equation for λ:

$$\begin{cases} M.A_t^x = B^x \\ M.A_t^y = B^y \\ M.A_t^z = B^z \\ A_c = M_g^{-1}.L^T.\lambda \\ \varepsilon.\ddot{s} = Ls^T.\lambda \\ L.M_g^{-1}.L^T.\lambda + \frac{Ls.Ls^T}{\varepsilon}.\lambda = E - L.A_t \end{cases}$$

We called n the number of control points of the spline so we have $M(n \times n)$ and $M_g(3n \times 3n)$. c is the number of constraints in the system, and thus is the height of L and Ls. And c_g is the number of new unknowns in the system, and thus is the width of Ls.

M is both constant over time and symmetric band matrix (cf. equation (6)) due to the spline locality property, we thus precompute an LU decomposition where L and U are two band matrices too. The computation time of A_t is then linear (thus in $\mathcal{O}(n)$).

We also can pre-compute the inverse matrix of M and use it to compute $M_g^{-1}.L^T$ then the matrix $R = L.M_g^{-1}.L^T$, this computation is in $\mathcal{O}(c.n^2 + c^2.n)$. We complete R by the integration of the sliding constraints part: $\frac{Ls.Ls^T}{\varepsilon}$. This completion is done in $\mathcal{O}(c_g.c^2)$.

Finally, we solve the matrix equation $R.\lambda = E - L.A_t$ which takes a complexity of $\mathcal{O}(n.c + c)$ for the right term computation, $\mathcal{O}(c^2)$ for R decomposition and $\mathcal{O}(c^2)$ for the final resolution. This gives us the values of the Lagrange multipliers and permits us to compute the correction accelerations $A_c = M^{-1}.L^T.\lambda$ (resolution in $\mathcal{O}(n.c)$) and the new value of the supplementary unknowns: $\varepsilon.\ddot{s} = Ls^T.\lambda$ (resolution in $\mathcal{O}(c.cg)$).

The trick employed in (13) permits to resolve the system in $\mathcal{O}(c.n^2 + c^2.n + c_g.c^2)$.

For comparison, a direct resolution gives a complexity of $\mathcal{O}((n + c_g + c)^3)$. By applying the same technique of acceleration decomposition on the unmodified system (12), this yields three different accelerations by dissociating the c constraints into c_1 usual constraints and c_2 smooth constraints ($c = c_1 + c_2$) [Rémion 2003]. This decomposition allows a resolution in the same order of complexity as the above resolution, but takes more computation stages. So the theoretical framework would be more time consuming.

6 Set of Smooth Constraints

Thanks to this free variables technique, a running knot can be easily defined using the constraint equation:

$$g(q, \dot{q}, t, s(t), s_0) = P(s(t), t) - P(s_0, t)$$

Such a constraint defines knot on a curve and also gives us the possibility to manipulate it easily, and change its position on the curve.

In the same way, a double sliding point constraint can be defined on a spline by the equation:

$$g(q, \dot{q}, t, s_1(t), s_2(t)) = P(s_1(t), t) - P(s_2(t), t)$$

This last constraint equation allows to manipulate a knot with the desired piece of spline. This gives us much more freedom on the manipulation but requires a second unknown.

Another application of this technique, is the reusing of the new free variables in others constraints equations. For example, we can imagine a sliding point constraint representing a surgical thread during a suture. At the contact point, the curve tangent can be constrained perpendicularly to the organ surface. This can be done by a tangent sliding point constraint equation:

$$g(q, \dot{q}, t, s(t), s_0) = \frac{dP(s(t), t)}{ds} - T(s_0, t)$$

where $T(s_0, t)$ is the tangent vector wanted at the $s(t)$ abscissa parameter.

We can imagine by the same way, a constraint on the normal vector:

$$g(q, \dot{q}, t, s(t), s_0) = \frac{d^2P(s(t), t)}{ds^2} - C(s_0, t)$$

with $C(s_0, t)$ is the curvature vector wanted at the $s(t)$ abscissa parameter.

Many constraint equations can be created by this way and new experimentation must be explored. Furthermore, this technique is practicable in the 2D case.

7 Generalization to surface

This formulation can be easily generalized to a 2D object, we just have a constraint based on the degrees of freedom. In this case, we would have two free variables for defining a sliding constraint. In terms of internal energies, we can apply some springs [Provot 1995] or trying a continuous energy [Terzopoulos et al. 1987; Nocent and Rémion 2001]. For a 2D spline (of size $n \times n$), we have the expression:

$$P(u,v,t) = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij}(t).b_i(u).b_j(v)$$

Then, the K term in the lagrangian equation gives us a much more complex mass matrix than the 1D case:

$$\frac{\partial K}{\partial q_{ij}} = (m. \sum_{i_1=1}^{n} \sum_{j_1=1}^{n} \int_0^1 \int_0^1 b_i(u) b_{i_1}(u) b_j(v) b_{j_1}(v) du dv) \dot{q}_{i_1 j_1}$$

As a result, M is composed of terms:

$$M_{(i_1 j_1),(i_2 j_2)} = m. \int_0^1 \int_0^1 b_{i_1}(u).b_{i_2}(u).b_{j_1}(v).b_{j_2}(v).du.dv$$

by adopting the coding line/column, a point of index i is the $q_{(i/n)(i\%n)}$ point, it gives a more precise formulation:

$$M_{ij} = m. \int_0^1 \int_0^1 b_{i/n}(u).b_{j/n}(u).b_{i\%n}(v).b_{j\%n}(v).du.dv$$

Such a matrix has a $n^2 \times n^2$ size (for a patch of size $n \times n$) and a band structure thanks to the spline locality that localizes the interactions between the control points in the 2D structure. A point interacts with some neighbors that are on the same line or on the neighbors line. These lines are localized in the matrix structure closely to the current one. In conclusion, we obtain a band matrix much larger than the 1D case.

Algorithms such as *Cuthill Mckee* also permit to re-organize the structure of the matrix by giving a different numeration of the knot[3]. It may enhance the resolution process by giving a more interesting structure to the M matrix.

A sliding point constraint for a surface is defined by the equation:

$$g(u(t),v(t),t) = P(u(t),v(t),t) - A$$

this constraint needs two free variables u and v to define a single sliding constraint. The computation will be more fastidious than the 1D case but the process is very similar.

Such constraint permits to define a point on the tissue through which the surface may slide while the user pulls it by an extremity. In a manner similar to a tissue that would slide on a stake if a person pulls it by one of its corner.

In term of complexity, the mass matrix grows to a $n^2 \times n^2$ size and always has a band property but with a larger band width. This remark set apart, the complexity is the same as in the 1D case.

8 Results

All the tests have been performed on a Pentium IV, 2.4GHz and 512Mb.

We first show that our method is effective with this animation example: A shoelace is pulled from one extremity (figure 1). The first image represents the initial situation where the shoelace is correctly put on the shoe, the holes are modeled using white spheres. The second picture shows what happens when we just pull it slowly. The shoelace slides along the sliding point constraints. The last picture represents the state of the dynamic process later when the shoelace passes through the two first holes. This last picture brings up the interesting property that each constraint can be activated or deactivated dynamically and automatically when the constraint becomes outside the curve, and can no longer be valid. The apparent rigidity of the shoelace is due to the discretization we chose for the spline (we could have a better movement by taking a much more subdivided spline, at the cost of a longer computation). In this example, the dynamic resolution process takes about 12ms for each computation step.

The second result shows the advantage of this method in regard of the distribution of the energy along the spline (figure 2). On the picture, the user manipulates the curve thanks to a probe which can be linked to a control point (which one can see in the second and third images). The crosses symbolize the control points of the spline in order to show their distribution. The first image is the initial situation where we take a spline on which we fix the extremities. Starting from this, we either fix its middle point (second picture) or define it as a sliding point (third picture). The result shows that the manipulation changes the modeling of the spline but the fixed point can be seen as two splines with constraint on the joining point, which localize the manipulation on one of the two sub-splines and some modification on the neighbor sub-spline due to the links constraint. For a sliding constraint, the spline is considered as a unique spline that we ask to pass through a specific point. The sliding process enables the spline to distribute its energy onto the entirely curve by passing the sliding point. This allows the control points to be evenly distributed along the curve.

The next example shows a slipknot constraint (figure 3). With such a constraint, we can make a knot with a spline and manipulate the spline in order to clamp the knot or on the contrary make it wider.

A useful application of this proposition is the direct manipulation of the sliding point constraint location. By this way, the model is accessible by its control points and also by its sliding points. Thus we impose that the curve passes through a point whose position is interactively set by the end-user(figure 4).

In figure 5, snapshots of sliding point constraints simulations on 2D spline are shown. The first image shows a 7×7 patch with three sliding points and one fixed point. The resolution process takes about 20ms for each computation step. The second image shows another configuration with the same patch but with only one sliding point constraint. For this example, the computation time is about 14ms.

In the different pictures, a green sphere represents a fixed point constraint and a white sphere represents a sliding point constraint.

9 Conclusion and Prospect

This article proposed a dynamic approach for the variational modeling and shared a specific constraint on a point without specifying explicitly the parameter of this point. With this type of constraint, we are able to make a direct edition and then offer a new way of modeling a spline.

This technique can be easily used in animation and could give interesting results for simulating all sorts of threads, ropes, etc.

It would be interesting to adapt the resolution of the spline curve depending on the constraints the user applies. Using multiresolution splines [Finkelstein and Salesin 1994] combined with automatic decision criteria, would be a valuable tool. Such a work is currently in progress.

[3] Boost Graph Library, Cuthill McKee Ordering. http://www.boost.org/libs/graph/doc/cuthill_mckee_ordering.html

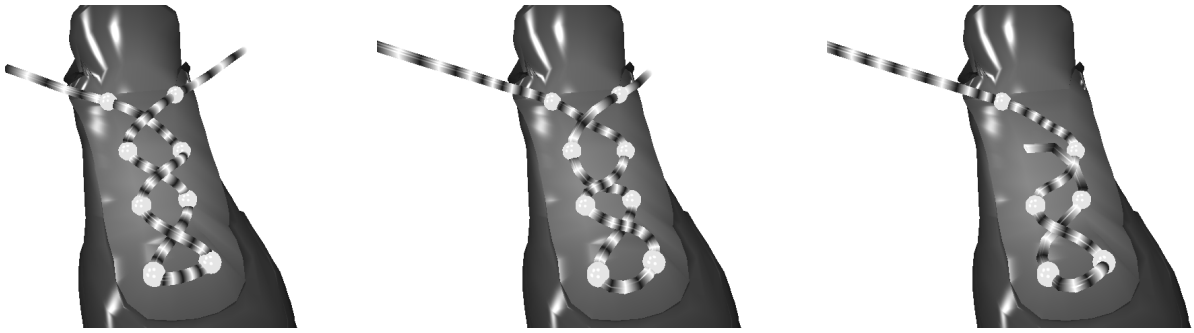

Figure 1: A shoelace sliding on some shoe hole (white spheres)

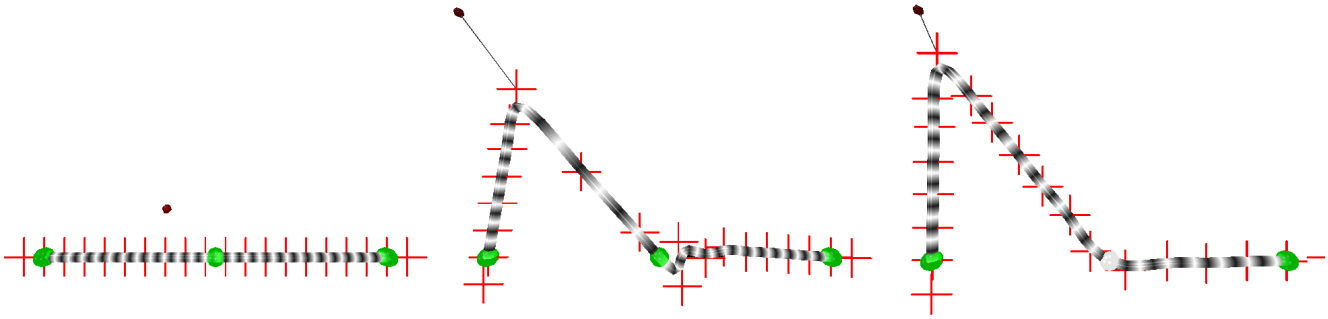

Figure 2: Left: Initial spline - Middle: Tension created with a fixed point
right: Correct re parameterization with a sliding point - (Control points are shown by the red crosses)

References

BARR, A. 1984. Global and local deformations of solid primitives. In *Computer Graphics (Proceedings of ACM SIGGRAPH 84)*, 18, 3, ACM, 21–26.

BAUMGARTE, J. 1972. Stabilization of constraints and integrals of motion. *Computer Meth. Appl. Mech. Eng. 1*, 1–16.

COQUILLART, S. 1990. Extended free-form deformation: A sculturing tool for 3d geometric modeling. In *Computer Graphics (Proceedings of ACM SIGGRAPH 90)*, 24, 4, ACM, 187–193.

FINKELSTEIN, A., AND SALESIN, D. H. 1994. Multiresolution curves. In *Proceedings of ACM SIGGRAPH 94*, ACM Press/ACM SIGGRAPH, New York., Computer Graphics Proceedings, Annual Conference Series, ACM, 261–268.

FORSEY, D., AND BARTELS, R. 1988. Hierarchical b-spline refinement. In *Computer Graphics (Proceedings of ACM SIGGRAPH 88)*, 22, 4, ACM, 205–212.

GLEICHER, M. 1992. Integrating constraints and direct manipulation. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, ACM, 171–174.

GORTLER, S. J., AND COHEN, M. F. 1995. Hierarchical and variational geometric modeling with wavelets. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, ACM, 35–42.

GRISONI, L., BLANC, C., AND SCHLICK, C. 1999. Hermitian b-splines. *Computer Graphics Forum 18*, 4 (Dec.), 237–248.

LENOIR, J., MESEURE, P., GRISONI, L., AND CHAILLOU, C. 2002. Surgical thread simulation. In *Proceedings of Modelling and Simulation for Computer-aided Medecine and Surgery (MS4CMS)*, EDP Sciences, Rocquencourt, vol. 12, INRIA, 102–107.

NOCENT, O., AND RÉMION, Y. 2001. Continuous deformation energy for dynamic material splines subject to finite displacements. In *Proceedings of the Eurographic workshop on Computer Animation and Simulation*, Springer Verlag, Manchester (UK), 87–97.

PIEGL, L., AND TILLER, W. 1997. *The NURBS Book*, second ed. Springer-Verlag, New York.

POTTMANN, H., LEOPOLDSEDER, S., AND HOFER, M. 2002. Approximation with active b-spline curves and surfaces. In *Proceedings of Pacific Graphics*, 8–25.

PROVOT, X. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behaviour. In *Graphics Interface'95 Conference*, 147–154.

QIN, H., AND TERZOPOULOS, D. 1996. D-NURBS: A Physics-Based Framework for Geometric Design. *IEEE Transactions on Visualization and Computer Graphics 2*, 1, 85–96.

RÉMION, Y., NOURRIT, J., AND GILLARD, D. 1999. Dynamic animation of spline like objects. In *Proceedings of the WSCG'1999 Conference*, 426–432.

RÉMION, Y. 2000. Animation dynamique : moteur lagrangien généraliste et applications. *Habilitation à diriger des recherches, Université de Reims, Champagne-Ardenne.* (Dec.).

RÉMION, Y. 2003. Prise en compte de "contraintes à variables libres". Tech. Rep. 03-02-01, LERI, Feb.

ROBINSON, S. M. 2003. Variational conditions with smooth constraints: Structure and analysis. In *International Symposium on Mathematical Programming*, 245–265.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISHER, K. 1987. Elastically deformable models. In *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, 21, 4, ACM, 205–214.

WELCH, W., AND WITKIN, A. 1992. Variational surface modeling. In *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, 26, 2, ACM, 157–166.

WITKIN, A., FLEISCHER, K., AND BARR, A. 1987. Energy constraints on parameterized models. In *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, 21, 4, ACM, 225–232.

ZORIN, D., AND SCHRODER, P. 2000. Subdivision for modeling and animation. *ACM SIGGRAPH 2000, course notes 23* (July).

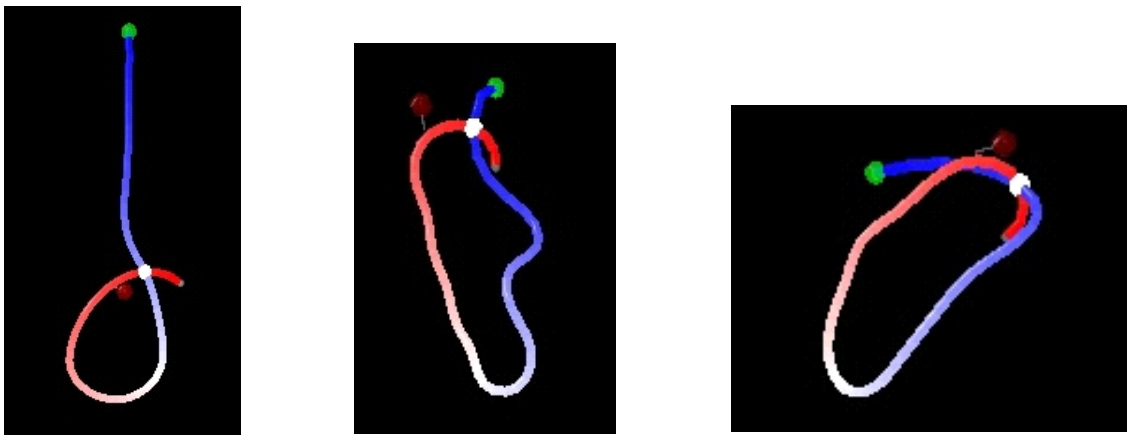

Figure 3: A rope for hang: a rope with a sliding point constraint linked to another point of the curve

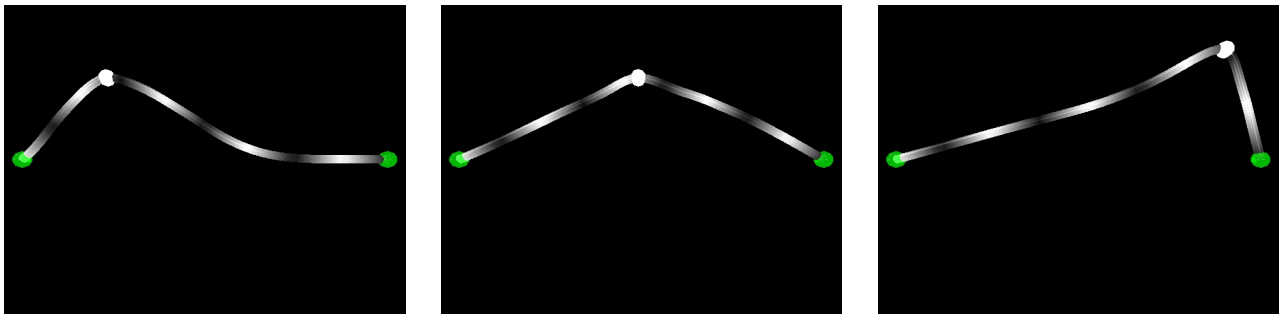

Figure 4: Direct Edition with an interactive sliding point constraint

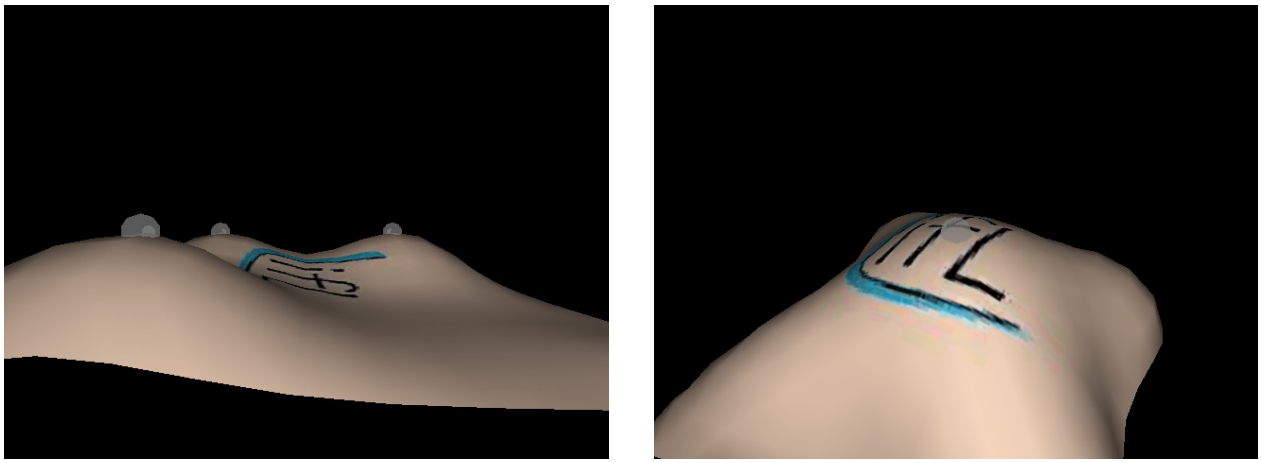

Figure 5: Application to 2D splines