www.cadanda.com
Computer-Aided Design
AND APPLICATIONS

Physically-Based NURBS Surface Editing With Curves

Yuewen Ma[1], Jianmin Zheng[2] and Kai Wang[3]

[1]Nanyang Technological University, mayu0005@ntu.edu.sg
[2]Nanyang Technological University, ASJMZheng@ntu.edu.sg
[3]Nanyang Technological University, kwang@pmail.ntu.edu.sg

ABSTRACT

This paper presents a physically-based approach for NURBS surface modification using curve handles. The approach consists of three steps: curve handle creation, target curve generation, and surface modification. The handle curve is a composite curve that is a composition of a NURBS surface and a 2D parametric curve defined in the parameter domain of the NURBS surface. The handle curve is treated as a physically-based dynamic curve whose formulation is derived directly based on the representation of the NURBS surface where the composition curve is defined. Then the target curve is generated as the result of dynamic evolution of the physically-based composition curve with external point forces. Finally, the initial handle curve and the target curve are used to define a curve force for the dynamic NURBS to modify the NURBS surface. The use of curves as shape handles makes the shape editing more effective and intuitive, and the incorporation of physics into geometric modeling makes the modeling process meaningful and the resulting surface smooth. Examples are provided to demonstrate the effectiveness and capability of the proposed NURBS manipulation method.

Keywords: NURBS, phyically-based modeling, shape manipulation, composite curves.
DOI: 10.3722/cadaps.2012.361-374

1 INTRODUCTION

Non-uniform rational B-splines (NURBS) are a popular mathematical representation for freeform curves and surfaces in computer aided design and manufacturing. There are many ways to edit NURBS curves and surfaces. For example, one may move control points, change weights, or adjust knots to modify the NURBS shapes. However, these methods are purely geometrically-based. Moreover, they lack the precision between the defining parameters and the shape, thus requiring understanding of the underling formulation.

Alternatively, physically-based dynamic NURBS (D-NURBS) provides an intuitive approach for NURBS sculpting [15]. Within D-NURBS, NURBS primitives are regarded as physical objects with mass distribution, elastic properties, etc. Besides using the NURBS control points and weights, users are allowed to interact with the objects through applying simulated forces and local and global constraints. Consequently, the shape deforms and evolves in a physically realistic manner in response to the interaction following Lagrangian Dynamics. Due to the physical nature, the deformation of NURBS objects becomes more intuitive and physically accurate.

While many modeling tools are point-based, curve handles for surface modification are becoming popular with the development of computerized sketching tools and are shown more effective and intuitive [8]. Especially when feature manipulation is required, curve handles are of greater advantage. This is because most surface features can be interpreted as curves on the surface. In NURBS domain, only a few works considered to use non-iso-curves for surface manipulation [7][16]. They mainly focused on B-spline surfaces. Basically, they specify some curves on the B-spline surface as shape handles, sketch some spatial curves as the target curves, establish the matching of the shape handles and the target curves, and then deform the surface. As curve handles provide a much more convenient way for the user to express his manipulation intention, such curve-based methods improve the easy-handling of surface manipulation. However, when coming to NURBS, things become complicated. The computation of explicit relation of control points and weights between the composite curve and the NURBS surface is difficult and time consuming. The specification of a general spatial target curve through sketching is challenging.

Based on these observations, in this paper we propose a physically-based approach for NURBS surface manipulation using general curves as handles. We first derive a formulation of physically-based dynamic composite curves directly on the representation of the NURBS surface where the composition curve is defined. Then we propose to use the dynamic composite curves as the shape handles and to generate the target curves from the composite curves using spring forces as the sculpting tool. Since springs can be placed in any location in 3D space, complicated target curves can be easily generated. Due to the use of physically-based model which inherently minimizes the internal energy, the deformation is smooth and natural. Moreover, in this way, the matching of the initial handle curve and the target curve is automatically established. Finally we use the initial handle curves and the target curves to define curve forces for D-NURBS and the NURBS surface editing is realized by the evolution of D-NURBS. The novelty of the paper is in aspects: (1) we formulate the dynamic composite curve in a direct way without deriving the explicit NURBS curve representation of the composite curve; and (2) we propose to edit NURBS surfaces using curve handles within a physically-based modeling framework.

The remainder of this paper is organized as follows: Related work and the D-NURBS basic are reviewed in Section 2 and Section 3, respectively. In Section 4, we present the formulation of physically-based dynamic composite curves, based on which a method for NURBS surface editing with curves is described in Section 5. Section 6 concludes the paper.

2 RELATED WORK

Aiming at enhancing the intuition and easy handling of NURBS surface modification, many efforts were made in the past two decades. One of the approaches for this purpose is direct manipulation. Direct manipulation specifies the desired shape's features such as point, curve or normal constraints directly and then the algorithm automatically converts these constraints into modification of corresponding NURBS shape parameters or degrees of freedom (DOFs). In [9], Piegl practised this idea to adjust the shape of NURBS surfaces, in which only one control point or weight was allowed to change. Hsu et al [4] proposed an approach that allows multiple points to move simultaneously. Flower [3] investigated how to manipulate B-spline surfaces through the first and second order derivatives. In [5], normal control was also discussed. In general, the direct manipulation methods basically convert the geometric constraints into equations in the NURBS DOFs. The deformed surface can be obtained by solving the equations. Sometimes an explicit solution may be available [9]. On the other hand, however, the direct manipulation often means local modification and only a few DOFs are involved. As shown in [5], when too few DOFs serve for modification, the final results tend to be unnatural or even have topological change.

Due to the pioneer works of Terzopoulos et al introducing deformable models into computer Graphics [14], many efforts were also made to marry physics with NURBS modeling, which produces physically-based modeling techniques. D-NURBS [15] is one of such examples. The idea of D-NURBS was also generalized to swung surfaces [12], triangular NURBS [13], and Hierarchical NURBS [18].

Celniker et al. [1] developed an interactive modeling system through minimizing thin-plate energy functional. Also making use of this functional, Celniker and Welch [2] investigated the deformable B-splines with linear constraints. Due to lack of local refinement in B-spline surfaces, Welch and Witkin [17] extended their work to trimmed hierarchical B-splines, where different constrained methods were considered. In [5], Hu et al. discussed manipulating NURBS surfaces through minimizing the thin-plate energy with extra normal control. Liu et al. [6] presented new methods to incorporate partial derivatives of an arbitrary order into the energy minimization models. In D-NURBS, if the variation of weights is allowed, the dynamic system becomes highly non-linear. To remedy this, Pourazady and Xu [10] provided a way to guide weight variation, which was formulated as a minimization in the homogeneous space. Since local modification is often desired in interactive modeling, Pusch and Samavati [11] proposed an approach that enabled local adjustment within the energy minimization.

3 D-NURBS SURFACES

A NURBS surface is defined as

$$\mathbf{S}(u,v)=\frac{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u)B_{j,l}(v)\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u)B_{j,l}(v)\omega_{i,j}},\tag{3.1}$$

where $\mathbf{P}_{i,j}$ are the control points, $\omega_{i,j}$ represent the weights, and $B_{i,k}(u)$ and $B_{j,l}(v)$ are B-spline basis functions of order k and l defined over knot vectors $\{u_0,u_1,\cdots,u_{m+k}\}$ and $\{v_0,v_1,\cdots,v_{n+l}\}$ respectively. The basis functions $B_{i,k}(u)$ can be written recursively by

$$B_{i,k}(u)=\frac{u-u_i}{u_{i+k-1}-u_i}B_{i,k-1}(u)+\frac{u_{i+k}-u}{u_{i+k}-u_{i+1}}B_{i+1,k-1}(u)\tag{3.2}$$

with the initial setting

$$B_{i,1}(u)=\begin{cases}1 & u_i\le u<u_{i+1}\\ 0 & \text{otherwise}\end{cases}.\tag{3.3}$$

The basis functions $B_{j,l}(v)$ can be written similarly. The control points form a control mesh. Fig. 1 shows a NURBS surface and the corresponding 6×6 control mesh.

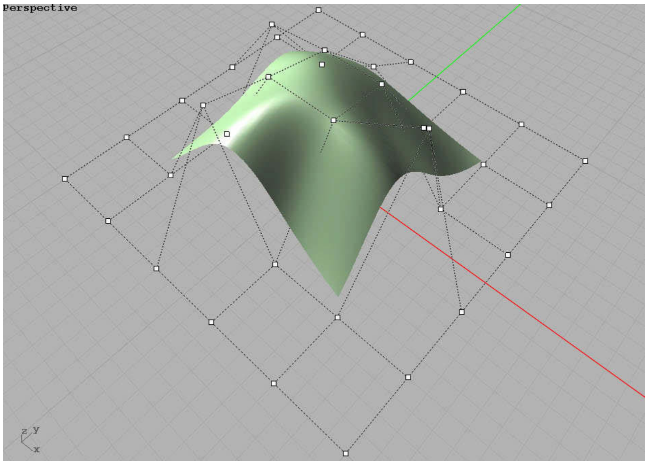


FIG. 1: A NURBS surface and its control mesh.

A D-NURBS surface is defined by

$$\mathbf{S}(u,v,t)=\frac{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u)B_{j,l}(v)\omega_{i,j}(t)\mathbf{P}_{i,j}(t)}{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u)B_{j,l}(v)\omega_{i,j}(t)}. \tag{3.4}$$

Compared to Eqn. (3.1), the control points and weights are now the functions of time t, which are also referred to as the generalized coordinates in a physical system. It is the variation of the generalized coordinates that produces the dynamic evolution of the D-NURBS surface. As a physical system, the generalized coordinates follow the Lagrangian dynamic equation:

$$\mathbf{M}\ddot{\mathbf{P}}+\mathbf{D}\dot{\mathbf{P}}+(\mathbf{K}+\lambda\mathbf{I})\mathbf{P}=\mathbf{f}_p-\mathbf{I}\dot{\mathbf{P}}+(\mathbf{K}+\lambda\mathbf{I})\mathbf{P}_0. \tag{3.5}$$

where the symbol • above a variable stands for taking the derivative of the variable with respect to time t, $\mathbf{P}$ is the concatenation of control points and weights

$$\mathbf{P}=\left[\mathbf{P}_{0,0}^{\mathrm{T}},\omega_{0,0},\mathbf{P}_{0,1}^{\mathrm{T}},\omega_{0,1},\cdots,\mathbf{P}_{m,n}^{\mathrm{T}},\omega_{m,n}\right]^{\mathrm{T}}, \tag{3.6}$$

$\mathbf{P}_0$ is $\mathbf{P}$ at the initial state, $\mathbf{M}$, $\mathbf{D}$, $\mathbf{f}_p$ and $\mathbf{K}$ represent the mass, damping, generalized force and stiff matrices:

$$\begin{aligned}\mathbf{M}&=\iint\mu(u,v)\mathbf{J}^{\mathrm{T}}\mathbf{J}\,du\,dv\\ \mathbf{D}&=\iint\gamma(u,v)\mathbf{J}^{\mathrm{T}}\mathbf{J}\,du\,dv\\ \mathbf{f}_p&=\iint\mathbf{J}^{\mathrm{T}}\mathbf{f}(u,v)\,du\,dv\end{aligned}$$

$$\mathbf{K}=\int\int\left(\alpha_{1,1}\mathbf{J}_u^{\mathrm{T}}\mathbf{J}_u+\alpha_{2,2}\mathbf{J}_v^{\mathrm{T}}\mathbf{J}_v+\beta_{1,1}\mathbf{J}_{uu}^{\mathrm{T}}\mathbf{J}_{uu}+\beta_{1,2}\mathbf{J}_{uv}^{\mathrm{T}}\mathbf{J}_{uv}+\beta_{2,2}\mathbf{J}_{vv}^{\mathrm{T}}\mathbf{J}_{vv}\right)dudv.$$

and

$$\mathbf{I}=\int\int\mu\mathbf{J}^{\mathrm{T}}\dot{\mathbf{J}}\,du\,dv.$$

Here $\mu(u,v)$ is the mass density distribution, $\gamma(u,v)$ is the damping density distribution, $\mathbf{f}(u,v)$ is the applied force distribution, the matrix $\mathbf{J}$ is the Jacobian matrix of surface $\mathbf{S}$ with respect to the generalized coordinates $\mathbf{P}$, and $\mathbf{J}_u$, $\mathbf{J}_v$, $\mathbf{J}_{uu}$, $\mathbf{J}_{uv}$ and $\mathbf{J}_{vv}$ are the first order and second order partial derivatives of $\mathbf{J}$ with respect to u, v or both. The expression of $\mathbf{J}$ is

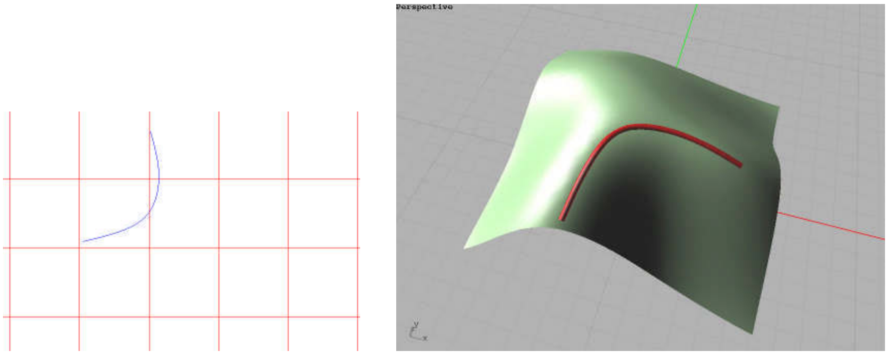
$$\mathbf{J}(u,v,t)=\left[\mathbf{B}_{0,0},\mathbf{W}_{0,0},\mathbf{B}_{0,1},\mathbf{W}_{0,1},\cdots,\mathbf{B}_{m,n},\mathbf{W}_{m,n}\right], \tag{3.7}$$

where $\mathbf{B}_{i,j}$ is a 3×3 diagonal matrix with the diagonal entries

$$N_{i,j}(u,v,t)=\frac{\partial\mathbf{S}}{\partial\mathbf{P}_{i,j}}=\frac{B_{i,k}(u)B_{j,l}(v)\omega_{i,j}}{\sum_{c=0}^{m}\sum_{d=0}^{n}B_{c,k}(u)B_{d,l}(v)\omega_{c,d}},$$

and $\mathbf{W}_{i,j}$ is the partial derivative of $\mathbf{S}$ with respect to $\omega_{i,j}$,

$$\mathbf{W}_{i,j}(u,v,t)=\frac{\partial\mathbf{S}}{\partial\omega_{i,j}}=\frac{\sum_{c=0}^{m}\sum_{d=0}^{n}B_{c,k}(u)B_{d,l}(v)\omega_{c,d}\left(\mathbf{P}_{i,j}-\mathbf{P}_{c,d}\right)}{\left(\sum_{c=0}^{m}\sum_{d=0}^{n}B_{c,k}(u)B_{d,l}(v)\omega_{c,d}\right)^2}.$$



(a) (b)

Fig. 2: A composite curve. (a) A 2D curve is defined in the parameter domain of a NURBS surface. (b) Mapping the 2D curve onto the NURBS surface gives a composite curve in 3D space.

Note that Eqn. (3.5) is a little different from the original equation in [15]. This is because we have modified the internal energy functional. Originally, the thin-plate energy is based on the absolute spatial coordinates. In physics, however, an object gains internal energy when the displacement away from its rest shape occurs. Thus we use the coordinate changes, rather than the absolute coordinates themselves, in the thin-plate energy functional. In addition, when we are in a modeling state of $\mathbf{P}_0$ where only minor modification is needed, we do not want the generalized coordinates to change too much during the evolution. So we add a penalty term into the energy functional:

$$\lambda(\mathbf{P}-\mathbf{P}_0)^{\mathrm{T}}(\mathbf{P}-\mathbf{P}_0)$$

where λ refers to the penalty coefficient, which can be adjusted to achieve different effects. With all the above setting, if we impose some forces into the D-NURBS, D-NURBS will deform in a natural and predictable way.

4 PHYSICALLY-BASED DYNAMIC COMPOSITE CURVES

This section extends the idea and approach of D-NURBS [15] to composite curves and presents physically-based dynamic composite curves or PBDC curves for short.

4.1 Dynamic Composite Curves

Consider a NURBS surface defined by Eqn. (3.1). In the parameter domain of the NURBS surface, there is a 2D curve G parameterized by s:

$$G=(u,v)=(u(s),v(s)).$$

Mapping the 2D curve onto the NURBS surface via substitution produces a composite curve $\mathbf{S}_c(s)$ in 3D space, which can be expressed as

$$\mathbf{S}_c(s)=\mathbf{S}(G)=\frac{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n}B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}}. \tag{4.1}$$

Fig. 2 is an example of such a composite curve.

Note that each NURBS control point or weight only affects a local region. Thus some control points or weights may not contribute to the composite curve. If we let Ξ denote the set of indices (i,j) of control points and weights that have influence on the composite curve, then the composite curve equation can be written

$$\mathbf{S}_c(s) = \frac{\sum_{(i,j)\in\Xi} B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}\mathbf{P}_{i,j}}{\sum_{(i,j)\in\Xi} B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}}. \tag{4.2}$$

In practice, the 2D curve G is often chosen to be a B-spline curve and then the composite curve is a 3D NURBS curve. The degree of the composite curve is generally very high. For instance, if G is a cubic B-spline curve, the composite curve has a degree of $3(k+l-2)$. A brutal force approach to defining the dynamic composite curve is to derive the NURBS representation of the composite curve first and then to apply D-NURBS curve formulation [15]. However, it is rather complicated to derive the NURBS representation of the composite curve from the equations of the NURBS surface and the 2D curve, and the degree of the resulting NURBS curve is very high.

Instead of deriving the explicit NURBS representation of the composite curve, we propose to formulate the dynamic composite curve directly on the composite representation:

$$\mathbf{S}_c(s,t) = \frac{\sum_{(i,j)\in\Xi} B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}(t)\mathbf{P}_{i,j}(t)}{\sum_{(i,j)\in\Xi} B_{i,k}(u(s))B_{j,l}(v(s))\omega_{i,j}(t)}. \tag{4.3}$$

That is, we use the NURBS surface control points as the generalized coordinates for the dynamic composite curve and they are functions of time t. By concatenating all these generalized coordinates into a vector

$$\mathbf{Q}(t) = \left[\cdots, \mathbf{P}_{i,j}^{\mathrm{T}}, \omega_{i,j}, \cdots\right]_{(i,j)\in\Xi}^{\mathrm{T}},$$

the dynamic composite curve can also be represented as

$$\mathbf{S}_c(s,t) = \bar{\mathbf{J}}\mathbf{Q}$$

where $\bar{\mathbf{J}}$ is the Jacobian matrix of $\mathbf{S}_c(s,t)$ with respect to $\mathbf{Q}$. Specifically,

$$\bar{\mathbf{J}}(s,t) = \left[\cdots, \mathbf{B}_{i,j}(u(s),v(s),t), \mathbf{W}_{i,j}(u(s),v(s),t), \cdots\right]_{(i,j)\in\Xi}$$

where $\mathbf{B}_{i,j}(u,v,t)$ and $\mathbf{W}_{i,j}(u,v,t)$ are the same as in Eqn. (3.7). If we let

$$\tilde{\mathbf{J}}(u,v,t) = \left[\cdots, \mathbf{B}_{i,j}(u,v,t), \mathbf{W}_{i,j}(u,v,t), \cdots\right]_{(i,j)\in\Xi},$$

which is a sub-matrix of $\mathbf{J}$, then $\bar{\mathbf{J}}$ is

$$\bar{\mathbf{J}}(s,t) = \tilde{\mathbf{J}}(u(s),v(s),t).$$

4.2 Physically-based Motion of Dynamic Composite Curves

We now impose mass and damping properties onto the composite curve, then Eqn. (4.3) simulates a physical object. In particular, let $\mu(s)$, $\gamma(s)$ and $\mathbf{f}(s)$ be the mass density, damping density, and applied force distributions, respectively, and the motion of the dynamic composite curve follow the work-energy version of Lagrangian dynamics. Using the similar derivation in [15], the dynamic equation governing the motion of generalized coordinates $\mathbf{Q}$ is

$$\bar{\mathbf{M}}\ddot{\mathbf{Q}} + \bar{\mathbf{D}}\dot{\mathbf{Q}} + (\bar{\mathbf{K}} + \lambda\mathbf{I})\mathbf{Q} = \bar{\mathbf{f}}_q - \bar{\mathbf{I}}\dot{\mathbf{Q}} + (\bar{\mathbf{K}} + \lambda\mathbf{I})\mathbf{Q}_0, \tag{4.4}$$

where the coefficient matrices are

$$\begin{aligned}
\bar{\mathbf{M}} &= \int \mu(s)\bar{\mathbf{J}}^{\mathrm{T}}\bar{\mathbf{J}}\,ds\\
\bar{\mathbf{D}} &= \int \gamma(s)\bar{\mathbf{J}}^{\mathrm{T}}\bar{\mathbf{J}}\,ds\\
\bar{\mathbf{f}}_p &= \int \bar{\mathbf{J}}^{\mathrm{T}}\mathbf{f}(s)\,ds\\
\bar{\mathbf{I}} &= \int \mu\bar{\mathbf{J}}^{\mathrm{T}}\dot{\bar{\mathbf{J}}}\,ds\\
\bar{\mathbf{K}} &= \int \left(\alpha\bar{\mathbf{J}}_s^{\mathrm{T}}\bar{\mathbf{J}}_s + \beta\bar{\mathbf{J}}_{ss}^{\mathrm{T}}\bar{\mathbf{J}}_{ss}\right)ds
\end{aligned} \tag{4.5}$$

and the derivatives of $\bar{\mathbf{J}}$ with respect to s are

$$\begin{aligned}
\bar{\mathbf{J}}_s &= \tilde{\mathbf{J}}_u(u(s),v(s),t)u' + \tilde{\mathbf{J}}_v(u(s),v(s),t)v',\\
\bar{\mathbf{J}}_{ss} &= \tilde{\mathbf{J}}_{uu}(u(s),v(s),t)u'^2 + 2\tilde{\mathbf{J}}_{uv}(u(s),v(s),t)u'v' + \tilde{\mathbf{J}}_{vv}(u(s),v(s),t)v'^2\\
&\quad +\tilde{\mathbf{J}}_u(u(s),v(s),t)u'' + \tilde{\mathbf{J}}_v(u(s),v(s),t)v''.
\end{aligned}$$

We call such a composite curve a physically-based dynamic composite curve.

Similar to D-NURBS, both linear and non-linear constraints such as the clamping of points or normals can be incorporated into PBDC curves. Also, for a PBDC curve, applied forces can serve as a sculpting tool for interactive editing of the curve. Particularly, the spring force is the most popular force. Fig. 3 shows two spring forces applied to a composite curve. To implement a spring force that connects a point $\mathbf{S}_c(s_0)$ on the curve to a spatial point $\mathbf{d}_0$, we just simply add

$$\int \bar{\mathbf{J}}^{\mathrm{T}}k(\mathbf{d}_0 - \mathbf{S}_c(s,t))\delta(s-s_0)\,ds$$

into the force term of Eqn. (4.4), where k is the spring stiffness coefficient and δ is the unit delta function.

Finally, it should be pointed out that the variation of weights may result in negative weights. To avoid this, we can impose extra weight constraints. For simplicity, we may just add a penalty term

$$c\left(\mathbf{q}_w - \mathbf{q}_w^0\right)^{\mathrm{T}}\left(\mathbf{q}_w - \mathbf{q}_w^0\right)$$

into the internal energy functional. Here $\mathbf{q}_w$ is the assembly of all the weights in $\mathbf{Q}$, $\mathbf{q}_w^0$ refers to the desired weights, and c is the penalty coefficient.

4.3 Numerical Implementation

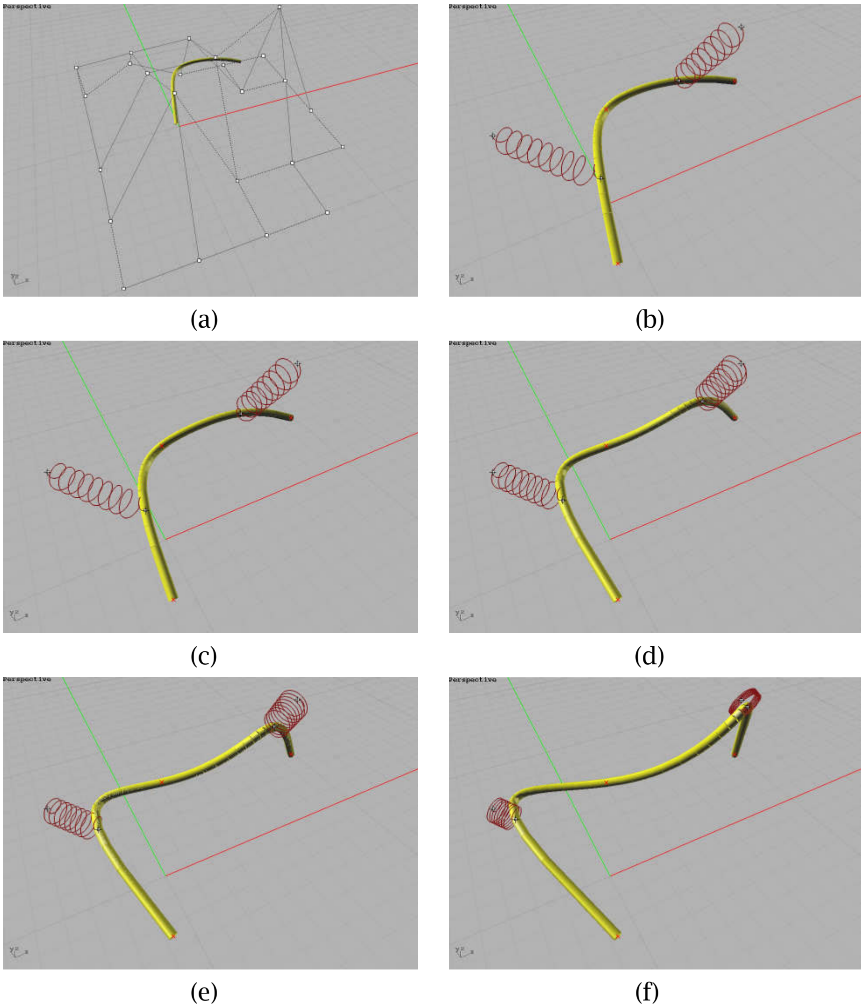
The dynamic behavior of a PBDC curve is governed by Eqn. (4.4). Due to the time-varying coefficient matrices, Eqn. (4.4) is a set of second order non-linear ordinary differential equations. In general, there is no analytical solution to Eqn. (4.4). Hence, we take a numerical approach to solve it using finite element and implicit integration methods.

To compute the coefficient matrices, we use the knots of both the surface and the domain curve to divide the curve into a set of segments and these curve segments are used as finite elements. Then for each element, Gaussian Quadrature is applied to numerically evaluate the integral. The final coefficient matrices are the sums of all element matrices.

To integrate the dynamic equation, the implicit integration method is employed, which is known to work numerically stably. First, the derivatives of the generalized coordinates $\mathbf{Q}$ are discretized over time steps Δt using backward difference:

$$\begin{aligned}
\dot{\mathbf{Q}}(t+\Delta t) &= \frac{\mathbf{Q}(t+\Delta t) - \mathbf{Q}(t-\Delta t)}{2\Delta t},\\
\ddot{\mathbf{Q}}(t+\Delta t) &= \frac{\mathbf{Q}(t+\Delta t) - 2\mathbf{Q}(t) + \mathbf{Q}(t-\Delta t)}{\Delta t^2}.
\end{aligned}$$

Then, the discretized derivatives are substituted into Eqn. (4.4) and we obtain a set of linear equations



(a) (b)

(c) (d)

(e) (f)

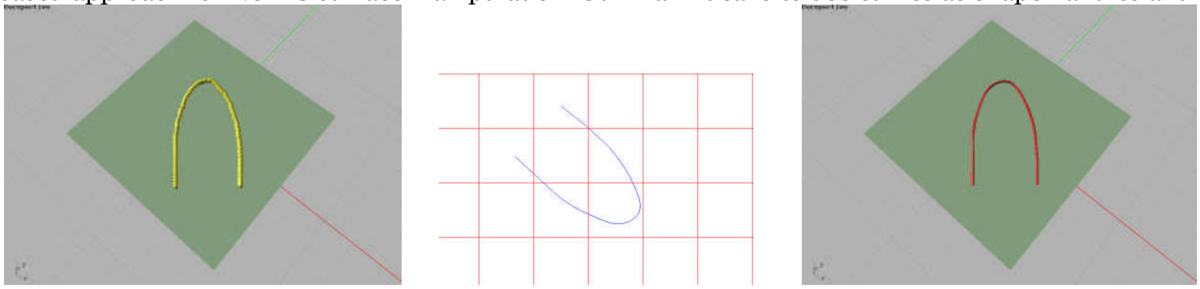
Fig. 3: Evolution of a PDBC curve. (a) Initial state. The white squares indicate the influential control points. (b)-(f) Dynamic evolution. The two ends of curve are fixed and the two spiral curves represent spring forces.

$$\begin{aligned}&\left(2\bar{\mathbf{M}}+\Delta t\bar{\mathbf{D}}+2\Delta t^2(\bar{\mathbf{K}}+\lambda\mathbf{I})\right)\mathbf{Q}(t+\Delta t)=\\&2\Delta t^2\left(\bar{\mathbf{f}}_q+(\bar{\mathbf{K}}+\lambda\mathbf{I})\mathbf{Q}_0)\right)+4\bar{\mathbf{M}}\mathbf{Q}(t)-(2\bar{\mathbf{M}}-\Delta t)\mathbf{Q}(t-\Delta t).\end{aligned}\tag{4.6}$$

Finally, solving the linear equations for $\mathbf{Q}(t+\Delta t)$ gives the state at $t+\Delta t$. Observing that the coefficient matrix of equation Eqn. (4.6) is symmetric, we use the conjugate gradient method to find the solution. In this way, the modeling with PBDC curve can be done at an interactive rate. Fig. 3 shows an example of evolution of a PBDC curves, where the initial state is the same as in Fig. 2.

5 PHYSICALLY-BASED MANIPULATION OF NURBS

After establishment of the PBDC curve framework, we are now in a position to describe our physically-based approach for NURBS surface manipulation. Our main idea is to use curves as shape handles and to deform curves and surfaces via the physically-based evolution. Our approach basically has three components: curve handle creation, target curve generation, and surface modification. Given a NURBS surface, the handle curve is defined as a composition of the NURBS surface and a 2D cubic B-spline curve defined in the parameter domain of the NURBS surface. The handle curve is then treated as a physically-based dynamic curve and the target curve is generated by sculpting the dynamic composition curve. Finally, the surface modification is accomplished by imposing the curve force, which is defined by the initial handle curve and the target curve, onto the D-NURBS surface. The details of these components are elaborated below.



(a) Stroke curve (b) domain curve (c) composite curve

Fig. 4: Curve handle creation.

5.1 Curve Handle Creation

Given a NURBS surface $\mathbf{S}(u,v)$ of Eqn. (3.1), we want to create a curve on the surface as shape handle. If an iso-parameter curve is required, it is simple to specify and create the curve. However, if we want to specify an arbitrary curve on the surface, it is not so simple. Similar to the approach in [7], we draw a stroke on the screen, and the stroke is mapped onto the NURBS surface and then mapped to the parameter domain of the NURBS surface. We approximate the pre-image of the stroke in the parameter domain by a cubic B-spline curve $G(s)$. Specifically,

$$G(s) = \sum_{i=0}^{M} B_{i,4}(s) q_i,$$

where q_i are the 2D control points. The composition of the NURBS surface $\mathbf{S}(u,v)$ and the curve $G(s)$ defines a 3D curve $\mathbf{S}(G(s))$ lying on the surface, which then serves as a curve handle.

Fig. 4 illustrates this process. A NURBS surface is shown in green. A stroke is drawn which is depicted in yellow in Fig. 4(a). A cubic B-spline curve is generated in the parameter domain of the NURBS surface is shown in Fig. 4(b), and the composite curve is shown in red in Fig. 4(c), which serves as a curve handle.

5.2 Target Curve Generation

While [7] sketches another 3D curve in space for the target curve, it is actually difficult to generate a smooth and fine-controllable target curve using sketching. We choose to generate the target curve by deforming the curve handle obtained in the preceding subsection. We regard the composite curve as a PBDC curve and introduce position constraints and spring forces to deform the shape of the curve. By applying spring force and point constraints, the PBDC curve evolves in a smooth way. During the evolution, we can also interactively sculpt the curve by changing the force. Compared with the sketching method in [7], the target curve can be more general and smoother. Fig. 5 demonstrates a progressive deformation.

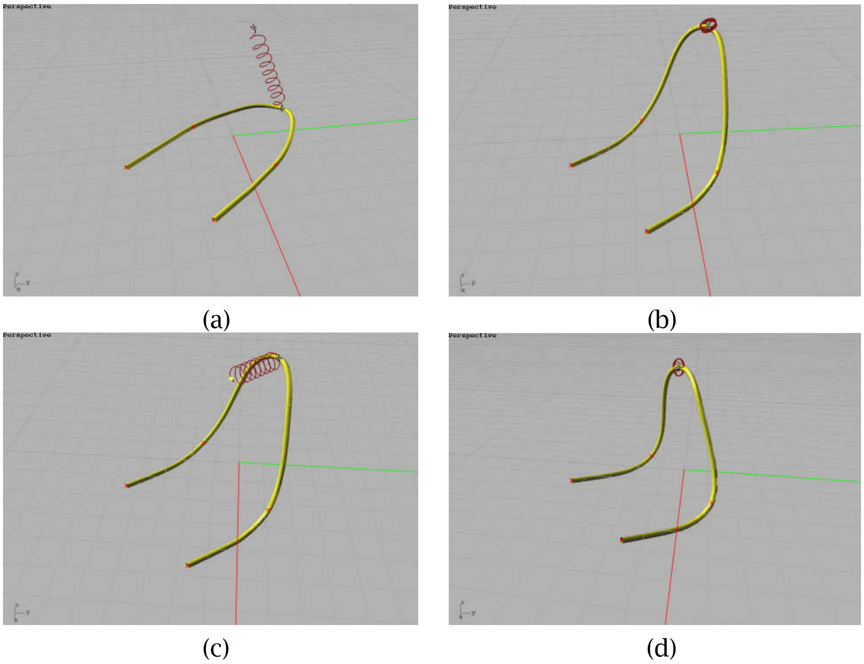


Fig. 5: Progressive deformation of a PBDC curve. (a)-(b) With a spring and 4 point clamping constraints, the central part of the PBDC curve deforms upwards. (c)-(d) With another spring, the central part of the PDBC curve further deforms to the left.

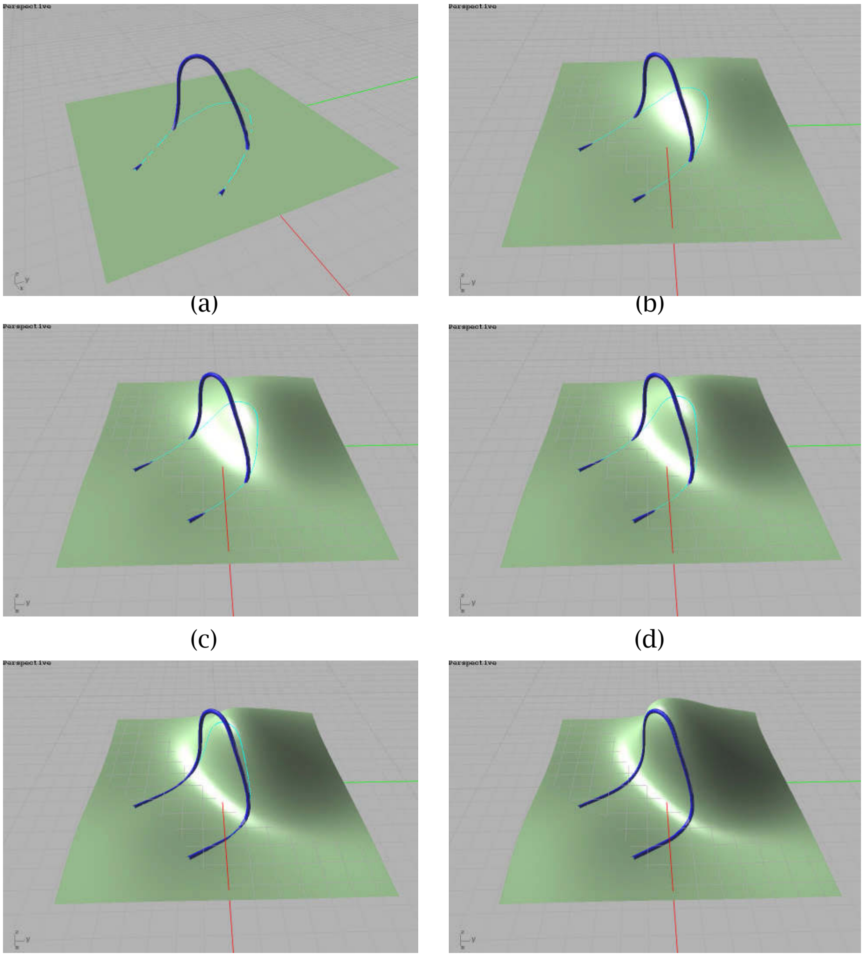
5.3 Surface Modification

After obtaining the desired target curve through the PBDC curve, what we need to do next is how to use such a curve for surface modification. We use the D-NURBS surface scheme for this purpose. Since the target curve is obtained from the curve handle, both curves have the same knot vector, number of control points, and parameterization, which provide a simple correspondence. Thus we define a curve force that connects the curve handle to the target curve. If we denote the target curve by $S_{c0}(x)$, the corresponding curve force is represented by

$$\int_{\mathbf{X}} k(s)\bar{\mathbf{J}}(u(s),v(s))^{\mathrm{T}} \left(\mathbf{S}_{c0}(s)-\mathbf{S}\left(u(s),v(s),t\right)\right)ds \qquad (5.1)$$

where $k(s)$ is the stiffness density along the curve. Actually Eqn. (5.1) simulates an infinite number of springs connecting the points of the handle curve to the corresponding points of the target curve, which pull the D-NURBS surface towards the target curve.

See Fig. 6 for an illustration. The curve handle (the sky blue line on the surface in Fig. 6 and the target curve (the dark blue cylindrical line in Fig. 6) that is obtained in Fig. 5(d) define the curve force. The curve force makes the D-NURBS gradually evolve from Fig. 6(a) to Fig. 6(f).



(a) (b)

(c) (d)

(e) (f)

Fig. 6: Dynamic evolution of the D-NURBS surface with curve force imposition.

5.4 Another Example

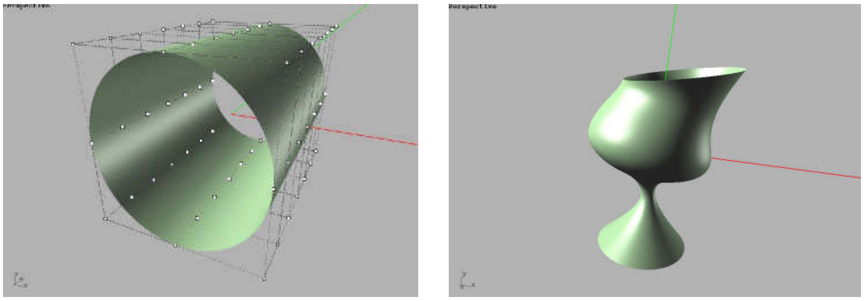
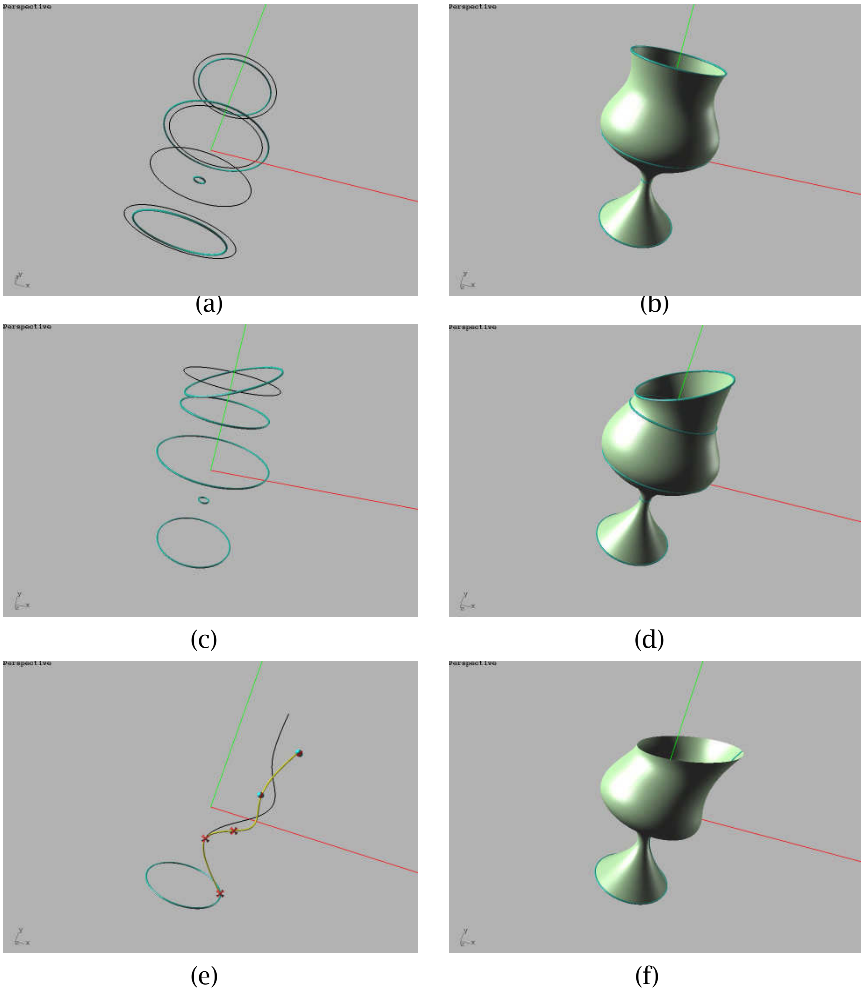


Fig. 7: Generate an asymmetric goblet from a cylinder. (a) Initial shape. (b) Final shape.



(a) (b)

(c) (d)

(e) (f)

Fig. 8: Progressive modeling of an asymmetric goblet from a cylinder.

Fig. 7 and Fig. 8 show an example of modeling an asymmetric goblet from a cylinder. The cylinder we use is a biquadratic NURBS surface. Fig. 8 displays the modeling steps of creating the goblet. In the first step (Fig. 8(a) and Fig. 8(b)), four iso-curves are chosen as handles and scaled to a desired dimension. After the first step, we obtain a conventional goblet. Next, we rotate the upper edge to impose a curve force and introduce extra four frozen curve constraints (Fig. 8(c) and Fig. 8(d)). After this step, the upper edge becomes askew while preserving the basic shape. Then we want to make the goblet bend to one side a little bit. To do this, we add a handling curve in the other direction to generate a different contour while the lower part of the curve handle is imposed as frozen constraint to maintain the base (Fig. 8(e) and Fig. 8(f)). Finally, a goblet with asymmetric contour is created.

6 CONCLUSION

This paper has described a new approach of using general curves as handles for NURBS surface modification. We first derive the formulation of physically-based dynamic composite curves directly in the composition representation. This avoids the computation of the explicit NURBS representation of the composite curve. Then we propose to use the physically-based dynamic composite curve as the handle curve and to generate the target curve from the handle curve in a physically-based sculpting.

After that, the NURBS surface modification is realized via the dynamic evolution of D-NURBS. Thus we practise the physically-based manipulation in two steps, which make the modeling process intuitive and natural and the deformation smooth and realistic. Finally, the use of general curves as handles enriches the modeling tools of D-NURBS and enhances the easy-handling of surface manipulation.

In future, how to speed up the integration of dynamic equations of both D-NURBS and PBDC curves and how to enhance the accuracy in the integration are important issues that warrant further study. As sketching tools are becoming popular, how to efficiently combine our method with sketching is also an interesting topic.

7 ACKNOWLEDGEMENTS

This work is supported by the ARC 9/09 Grant (MOE2008-T2-1-075) of Singapore.

REFERENCES

[1] Celniker, G.; Gossard, D.: Deformable curve and surface finite-elements for freeform shape design, Proceedings of the 18th annual conference on Computer graphics and interactive techniques, ACM, NY, USA, 25(4), 1991, 257-266. DOI: 10.1145/122718.122746

[2] Celniker, G.; Welch, W.: Linear constrains for deformable B-spline surfaces, Proceedings of the Symposium on Interactive 3D Graphics, Computer Graphs, Boston, 25(2), 1992, 165-70. DOI: 10.1145/147156.147191

[3] Fowler, B.: Geometric manipulation on tensor product surface, Proceedings of the 1992 Symposium on Interactive 3D Graphics Cambridge, ACM, 1992, 101-108. DOI: 10.1145/147156.147172

[4] Hsu, W.M.; Hughes, J.F.; Kaufmann, H.: Direct manipulation of free-form deformation, Computer Graphics, 26(2), 1992, 177-184. DOI: 10.1145/133994.134036

[5] Hu, S.-M.; Li, Y.-F.; Ju, T.; Zhu, X.: Modifying the shape of NURBS surfaces with geometric constraints, Comput. Aided Design, 33(12), 2001, 903-912. DOI: 10.1016/S0010-4485(00)00115-9

[6] Liu, X.; Feng, F.: Energy-based shape modification of NURBS surfaces, In Proceedings of the 2009 First International Workshop on Education Technology and Computer Science, IEEE Computer Society, Washington, DC, USA, 01, 2009, 929-932. DOI: 10.1109/CSO.2009.113

[7] Michalik, P.; Kim, D. H.; Bruderlin, B. D.: Sketch- and constraint-based design of B-spline surfaces, Proceedings of the seventh ACM symposium on Solid modeling and applications, ACM, 2002, 297-304. DOI: 10.1145/566282.566325

[8] Nealen, A.; Igarashi, T.; Sorkine, O.; Alexa, M.: FiberMesh: designing freeform surfaces with 3D curves, ACM Trans. Graph., 26(3), 2007. DOI: 10.1145/1276377.1276429

[9] Piegl, L.: Modifying the shape of rational B-splines. Part 2: surfaces, Computer-Aided Design, 21(9), 1989, 538 - 546. DOI: 10.1016/0010-4485(89)90014-6

[10] Pourazady, M.; Xu, X: Direct manipulation of NURBS surafces subjected to geometric constraints, Computer & Graphics, 30(4), 2006, 598-609. DOI: 10.1016/j.cag.2006.03.012

[11] Pusch, R.; Samavati, F.: Local constraint-based general surface deformation, Shape Modeling International Conference, 2010, 256-260. DOI: 10.1109/SMI.2010.39

[12] Qin, H.; Terzopoulos, D.: Dynamic swung surfaces for physics-based shape design, Computer Aided Design, 27(2), 1995, 111-127. DOI: 10.1016/0010-4485(95)92151-H

[13] Qin, H.; Terzopoulos, D.: Triangular NURBS and their dynamic generalizations, Computer Aided Geometric Design, 14(4), 1997, 325-347. DOI: 10.1016/S0167-8396(96)00062-3

[14] Terzopoulos, D.; Platt, J.; Barr, A.; Fleischer, K.: Elastically deformable models, Computer Graphics, 21(4), 1987, 205-214. DOI: 10.1145/37402.37427

[15] Terzopoulos, D.; Qin, H.: Dynamic NURBS with geometric constraints for interactive sculpting, ACM Transactions on Graphics, 13, 1994, 103-136. DOI: 10.1145/176579.176580

[16] Wang, Z.-G; Zhou, L.-S.; Wang, X.-P.: Direct Manipulation of B-spline surface, Chinese Journal of Mechanical Engineering, 18(1), 2005, 103-108. DOI: 10.1145/133994.134036

[17] Welch, W.; Witkin, A.: Variational surface modeling, Proceedings of 19th annual conference on Computer Graphics and interactive techniques, ACM, NY, USA, 1992, 157-166. DOI: 10.1145/133994.134033

[18] Zhang M.; Qin, H.: Hierarchical D-NURBS surfaces and their physics-based sculpting, Proceedings of the International Conference on Shape Modeling & Applications, IEEE Computer Society, Washington, DC, USA, 2001, 257. DOI: 10.1109/SMA.2001.923397