EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2011)
T. Hammond and A. Nealen (Editors)

# Sketch-Based Modeling of Smooth Surfaces Using Adaptive Curve Networks

Günay Orbay and Levent Burak Kara

Carnegie Mellon University, Pittsburgh, PA, USA

**Abstract**

*We present a new 3D surface modeling method that enables a rapid creation and modification of globally smooth surfaces from curve networks. The key feature of the proposed method is that it assumes the sketched curve networks to be malleable rather than rigid, thus enabling a mediation between curve interpolation versus surface smoothness. In the first step, the user sketches a network topology in the form of cubic feature edges. The curve network serves as an initial control mesh, from which a subdivision surface is computed. The subdivision surface is then iteratively modified to make the limit surface interpolate the original curve network at an arbitrary number of points, while minimizing the curvature variation energy of the surface. For networks in which this forced interpolation causes undesirable distortions to the surface, the network is automatically adjusted to make it conform to a smoothed version of the surface. This approach enables a concurrent modeling of the curve network and the underlying surface, thus eliminating the need for a laborious, iterative adjustment of the curve network for smooth surface creation.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computer Geometry and Object Modeling—Modeling packages, I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

## 1. Introduction

A variety of sketch-based interfaces systems have been developed over the years for geometric modeling applications in computer graphics and computer aided design [OSSJ09, JS11]. Particularly in free-form surface modeling, many such systems seek a fast transformation from 2D to 3D through simple object instantiation, followed by an iterative modification of the geometry with pen strokes [NISA07, TZF04, SWSJ06, KH06, DAJ03]. Alternatively, a number of systems have been proposed that involve primitive sketching, followed by gestural interactions that implement common CAD operations [OSSJ09]. A common feature in both approaches is that the sketched strokes are used either to define an initial starting shape such as a contour or silhouette, or to progressively modify an existing shape.

In this work we describe a surface modeling approach that works from 3D networks of curves that are progressively sketched from arbitrary perspective views. Compared to previous approaches, this approach allows users to sketch the characteristic feature curves of their design, similar to the way product designers explore shape ideas with pencil and paper sketches [ES07]. This provides more latitude in the types of models that can be created, as the input strokes are not required to form contours or silhouette lines to be converted into 3D primitives. From a user interaction stand point, our approach is closely related to ILoveSketch [BBS08] in which 3D wireframe models are constructed with 2D strokes. The key advance is that our approach allows smooth surface construction on wireframes that can be interactively visualized, progressively developed, and edited using the sketch input.

Figure 1 shows a typical usage of our system. Users sketch wireframe models with arbitrary topologies and the system produces interpolating surfaces. The proposed approach seeks to achieve two specific goals: (1) Users create and interact with the model solely through the sketched curve network, (2) Computed surfaces are to be at least $G^1$ continuous but $G^2$ wherever possible, without topology limitations.

© The Eurographics Association 2011.

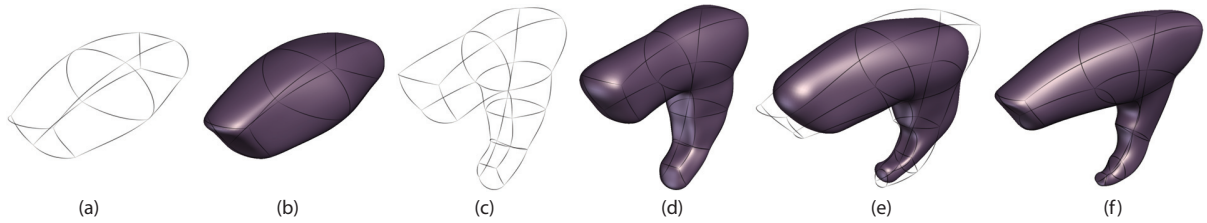


**Figure 1:** *Usage of our system. Curve network construction followed by surfacing, curve topology modification, surface updating, curve editing, and final surface creation.*

## 2. Overview

A key challenge is the generation of a smooth surface given an arbitrary curve network. For a curve network to represent a smooth and continuous surface, its constituent curves need to satisfy certain geometric constraints. For instance, all curves must be tangent to the underlying surface, which requires incident curves to share a common surface normal at their junctions [HPS10]. Such constraints are not readily satisfied. Figure 2 illustrates a case in which the three connecting curves do not form a common surface normal. Such situations occur frequently in sketch-based interfaces as it is difficult to configure the curves to share a common surface normal in the absence of an underlying surface.

Three immediate possibilities exist to remedy this issue:

- An interpolating surface (possibly parametric) can be instantiated, requiring explicit normal and twist control at the junctions for smoothness (Figure 2.b).
- A high-poly mesh can be used to represent the surface. While flexible, this representation may require energy functionals to be minimized on the whole mesh, and the mesh topology may have to be recalculated after each stroke [OSSJ09].
- Subdivision surfaces can be created that are naturally smooth. However, these surfaces may lie significantly far from the curve network, thus creating a discrepancy between the feature edges and the resulting surfaces (Figure 2.c and d).

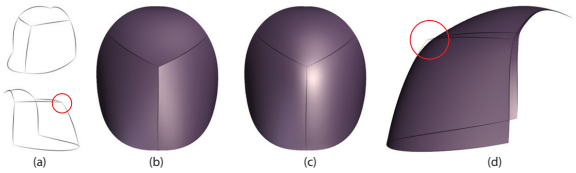


**Figure 2:** *(a) A curve network which does not result in a smooth surface. (b) Interpolating Cubic Coons patches do not generate smooth transitions across the patches. (c) and (d) A subdivision surface is smooth, but does not interpolate the curve network.*

In this work, we seek a solution that is a variation of the third possibility, where we exploit the smooth limit surfaces of the Catmull-Clark subdivision scheme [CC78]. Specifically, we use the sketched curve network to generate a set of bicubic patches that are used for computing an initial control mesh for subdivision. The resulting limit surface typically lies far away from the curves. As a solution, we propose an iterative update scheme that allows the subdivision surface to interpolate the curve network at a desired number of points along each edge. During this process, a curvature variation minimizing energy functional is applied to the coarsest-level control mesh, thereby producing a globally smooth surface. The resulting surface is $G^2$ continuous across the ordinary (4-valence) vertices of the sketched network, and $G^1$ otherwise.

We also provide a means for geometric negotiation between the curve network and the resulting surface. Rather than treating the input network as a rigid curve set, we allow the network to conform to the underlying surface when desired. We have found this capability to be useful in situations where the computed surface has undesirable artifacts that arise from the network geometry. In such situations, the user can interactively relax the surface energy, which simultaneously adjusts the curves with the surface. Figure 11 illustrates the idea and will be discussed later.

## 3. Related Work

A large body of sketch-based 3D modeling techniques have been presented to date. [OSSJ09, JS11] provide surveys of many methods. Closely related to our work are the sketch-based blob creation and modification methods, as well as mesh editing methods [NISA07, TZF04, SWSJ06, NSACO07]. Fibermesh [NISA07], for instance, produces a curvature minimizing closed mesh that interpolates user-drawn contour and edge curves. Typically, the first stroke is a closed curve that initiates a blob that is later used as a template that receives the subsequent control curves. With each new curve, the mesh structure is updated and a non-linear system of vertex coordinates is solved to produce the surface shape. This approach is well-suited to creating voluminous organic shapes where sketched curves represent circumscribing contours, blobby features, and the sharp or soft borders between different surface features. Our approach shares a similar goal of energy minimizing sur-

© The Eurographics Association 2011.

faces [MS92, WW94, YFSH99]. However, we utilize subdivision surfaces where the sketched curves loosely represent a coarse-level control mesh. We believe our approach is better suited to the kinds of geometries encountered in product design.

Works of Nasri *et al.* [NKS09] and Bein *et al.* [BHSF09] are also of interest to our approach. Nasri *et al.* [NKS09] uses polygonal complexes to create models that have user-defined outer contours, followed by subdivision for surfacing. However, the produced geometries are limited to flat inflated models. Bein *et al.* [BHSF09] present sketch-based tools for creating subdivision surfaces using operators such as extrusion and revolution. The modifications to the models are done through vertex/edge additions and vertex/face dragging. Although a useful tool, this method only utilizes sketch input to draw shapes on prescribed construction planes. The proposed work, however, uses sketch input for direct 3D curve design and for curve-level interaction with the created surfaces.

Also related to our approach, previous works include methods for free-from surfacing of curve networks. Levin [Lev99] proposed a combined subdivision scheme to interpolate networks of curves given that no more that curves intersect at one point. This constraint limits the input curve networks to specific topologies that can be interpolated. Schaefer *et al.* [SWZ04] proposed a modified subdivision scheme for interpolation networks of polylines which converge to cubic bsplines on the limit surface. Although capable of producing smooth and creased surfaces, the produced surface patches are manually defined by the user. Das *et al.* [DDGG05] developed a free-form surfacing method that works from 2D sketches of network of curves. The resulting surface meshes interpolate the estimated 3D curves, however, they are not suitable to efficiently produce $G^2$ geometries. In contrast, the proposed work aims for a fluid design environment where the sketched curve networks are quickly and automatically skinned with smooth surfaces.

Kara and Shimada's SketchCAD system [KS07] uses template surfaces to project input strokes and instantiate 3D curves. Each surface patch is created independently, requiring the user to manually establish geometric continuities higher than $G^0$. In contrast, the proposed work does not require a template surface as it is capable of directly creating curves in 3D space. The geometric continuity across neighboring patches is established automatically as an inherent property of the subdivision surfaces. In addition, the mesh complexity of the resulting limit surfaces can be efficiently computed at any resolution using parametric approximations of subdivision surfaces [LS08].

Bae *et al.* 's ILoveSketch [BBS08] presents a template-free, curve modeling system that allows symmetric 3D models to be constructed from arbitrary viewpoints with 2D strokes. In our approach, we use the single view epipolar sketching method for network construction, but later create subdivision surfaces on the network. Also, our sketch-based curve editing method allows individual curves to be modified arbitrarily, thus allowing construction of non-symmetrical final models. Of similar interest is the analytic sketching method proposed by Schmidt *et al.* [SKSK09] where construction lines and planes are sketched exploiting the vanishing points specified by the user. While we do not use it in this work, the analytic drawing can be used for 3D curve network construction.

## 4. Technical Details

In this section, we describe the three major components of our system: (1) 3D curve design, (2) Automatic loop identification and surface patch creation, (3) Surface and curve network updating to adjust interpolation/smoothness.

### 4.1. 3D Curve Design

As reviewed earlier, previous studies have introduced a number of 3D curve creation techniques using pen input. A recent work [BBS08] compiled a number of methods in a system and demonstrated the fluidity of the curve design process. In this work, we use a subset of these methods to instantiate and modify the curve networks. Specifically, our system uses the single-view symmetric epipolar method to create the initial curves. This process is at the heart of the curve-level interaction between the user and our system. Once created, each curve can be individually modified in space by over-sketching.

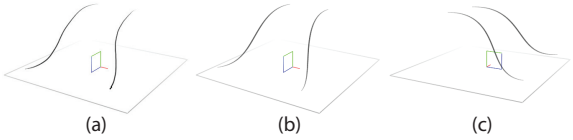


**Figure 3:** *(a) The user draws symmetric curve pairs. A least squares solution situates the two curves in space. (b,c) The created curves from different view points.*

The interface of our system shows a horizontal and a perpendicular plane demarcated with Cartesian axes. The vertical plane is assumed to be the symmetry plane. The user sketches a pair of curves that will be symmetrically oriented across the symmetry plane (Fig. 3). The 3D positions of the curves are then calculated using the symmetric epipolar method. Once created, existing curves can be modified by sketch input. In this mode, the user selects a curve and draws its new shape from any desired view point (Fig. 4). The new shape of the 3D curve is computed such that the curve goes through minimum amount of deformation. The curve network is then trimmed to form closed loops by attaching the curves at their end joints with a lasso tool. This results in the closed curve networks similar to those shown in Fig. 1.

© The Eurographics Association 2011.

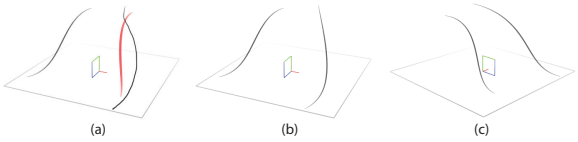


**Figure 4:** *(a) Similar to curve creation, the user sketches a modifying curve. (b,c) The modified curve from different view points. In this example, the user chooses not to preserve the symmetry during modification.*

### 4.2. Automatic Loop Identification for Surfacing

The individual surface patches are automatically computed from the curve network similar to [SL96]. However, to facilitate efficient search for candidate loops, we use existing graph matching methods. For this, the curve connectivity stored in the joint data structure is used to encode the curve network as an undirected graph. A graph matching algorithm is then used to automatically identify all candidate curve loops that may form a surface patch. The curve loops that invalidate the 2-manifold structure are iteratively eliminated until no such loop exists. The remaining loops are then used to construct the surface patches which subsequently facilitate subdivision. Our techniques, which are described in the following paragraphs, are currently implemented for four-sided loops only. However, these techniques are general enough to be easily extended to n-sided loops.

#### 4.2.1. Graph Representation of Curve Networks

In the graph representation, we encode both the curves and the joints formed by these curves as graph nodes. The main reason behind this encoding is that with this representation, the joints that are connected to one another through multiple curves can be properly distinguished, which would not be possible with a simpler encoding (*e.g.* where nodes and edges denote joints and curves, respectively). An example encoding is shown in Fig 5.b where the joints and curve nodes are shown with hollow and solid circles, respectively.

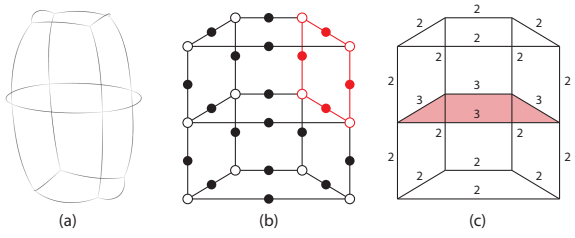


**Figure 5:** *(a) The curve network converted into (b) a graph structure where hollow and solid nodes denote curves and joints respectively. Four-sided loops (in red) are identified using graph matching. (c) The number of adjacent loops is used to eliminate the loops that invalidate the 2-manifold structure.*

#### 4.2.2. Graph Matching for Loop Search for Surfacing

We use subgraph isomorphism to identify the salient four-sided curve loops in the curve network. We define the graph matching problem by creating a small graph representing a four-sided loop. This query graph is then searched topologically within the curve network graph. This query yields all candidate loops including multiple copies of the same curve loop due to the rotational symmetry of the four-sided curve loop. That is, for the curve loop A-B-C-D, loops B-C-D-A and D-C-B-A also appear as different matches. Before proceeding, our system removes such duplicates.

In many cases, there exists a number of loops which invalidate the 2-manifold property of the resulting surface. Eliminating these loops requires an analysis of the possible edges where the 2-manifold structure is violated. Our system identifies such edges by counting the number of loops which share the same edge (an example counting is shown in Fig. 5.c). Starting from the loop that has the highest number of offending edges, our system iteratively distills the initial set of candidates until no such loop exists in the set.

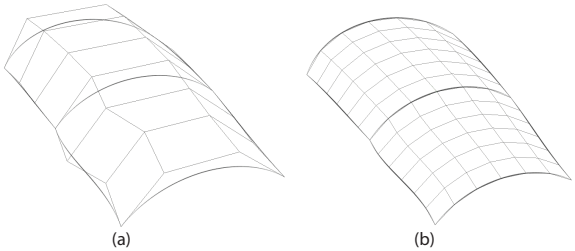


**Figure 6:** *(a) The control points of a bicubic Coons patch computed from the curve network. (b) A initial control mesh for subdivision is computed by sampling points from the Coons patch. Note that the control mesh initially interpolates the curve network.*

The resulting four-sided curve loops are next converted to subdivision surface patches. We begin by fitting bicubic Bezier Coons patches [Coo67] on each curve loop. We then sample points on the parametric surface to initiate the control vertices for the coarsest level subdivision patch. The sides of each patch are divided into a user specified number of control polygon edges.

### 4.3. Surface and Curve Network Updating

#### 4.3.1. Subdivision Limit Surface Analysis

Catmull-Clark subdivision surfaces are generalization of bicubic bspline surfaces to arbitrary topologies, thus has similar properties. Although they produce smooth, $G^2$ continuous ($G^1$ at extraordinary vertices †) surfaces, they do not

† The vertices where the quad mesh is not regular, that is, number of incident edges is not equal to 4.

© The Eurographics Association 2011.

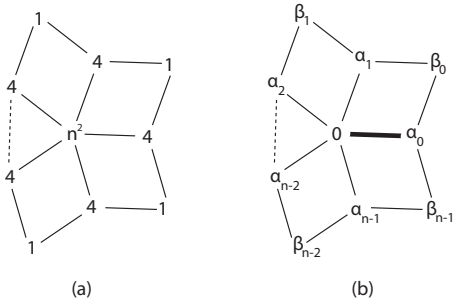


**Figure 7:** *The Catmull-Clark masks for (a) limit positions and (b) limit tangents along an edge shown in bold.*

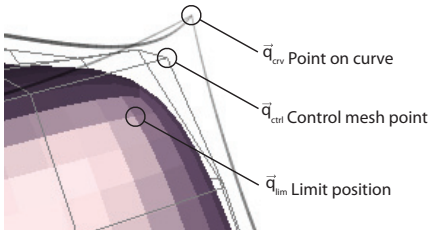


**Figure 8:** *The subdivision surface is iteratively updated to draw the limit surface onto the curve network.*

necessarily interpolate the vertices of their control meshes. Halstead *et al.* [HKD93] presented a detailed analysis on the geometric properties of Catmull-Clark limit surfaces. They also presented vertex weight masks to be used for calculating limit surface positions and tangents at locations corresponding to the control mesh vertices at the coarsest level. Figure 7 shows these weight masks for computing the limit position at a vertex, and the limit tangent in the vicinity of a vertex along an edge. $\alpha$ and $\beta$ used for limit tangent calculation are defined as follows:

$$\alpha_i = \cos\left(\frac{2\pi i}{n}\right) \quad (1)$$

$$\beta_i = \left(\frac{\sqrt{4+\cos(\frac{\pi}{n})^2}-\cos(\frac{\pi}{n})}{4}\right)\cos\left(\frac{2\pi i+\pi}{n}\right) \quad (2)$$

where $n$ is the valence of the vertex in question. For the limit position mask to yield the correct position, it weights should be normalized to sum up to 1. These numbers define the weighting scheme in the one-ring neighborhood of a vertex. For instance, the limit position of a vertex is calculated as the weighted average of the positions of that vertex and the vertices in its one-ring neighborhood. Likewise, the limit tangent mask is oriented and applied to the one-ring neighbors. In the following sections, we heavily make use of both of the limit positions and limit tangents.

#### 4.3.2. Iterative Surface Interpolation and Minimization of Curvature Variation

As described earlier, designing a globally smooth surface which interpolates a given curve network is not trivial. One difficulty lies in the fact that not all curve networks are readily interpolated by a smooth surface. The other difficulty is that the geometry of the regions far from the curves are, by definition, initially undetermined. In this step, we iteratively update the coarsest level vertices of the subdivision surface control mesh such that its limit surface interpolates the curve network at a set of prescribed points, while it approximates elsewhere. At the same time, the variation of curvature on the control mesh is minimized for the interior regions.

As shown in Fig. 6, the vertices of the control mesh corresponding to the curve network initially lie precisely on the network (since Coons patches interpolate their edges). The limit surface, however, is typically farther away from the control mesh. Hence, the goal is to adjust the control mesh such that the limit surface moves closer to the curve network. For an initial control mesh consisting of $N$ edges per sketched curve (the user can control this number), vertex $n$ ($n = 0, 1...N$) of the control mesh is linked with the position on the sketched curve corresponding to the parametric coordinate of $u_n = n/N$. For every control mesh vertex, the corresponding limit surface position is also known from the Catmull Clark masks described earlier. The difference between the limit positions of the vertices and the curve network is thus iteratively minimized by updating the control polygon vertices as follows:

$$\vec{q}_{ctrl}^{\,t+1} = \vec{q}_{ctrl}^{\,t} + \lambda(\vec{q}_{crv} - \vec{q}_{limit}^{\,t}) \quad (3)$$

where $\vec{q}_{ctrl}^{\,t+1}$ is the updated position of the control vertex, $\vec{q}_{crv}$ is the position on the curve at the corresponding parameteric coordinate, and $\vec{q}_{limit}^{\,t}$ is the limit position of the control vertex $\vec{q}_{ctrl}^{\,t}$. $\lambda$ is a damping factor between $[0,1]$ used during iterative updating. In our implementations, we used 0.5 for $\lambda$.

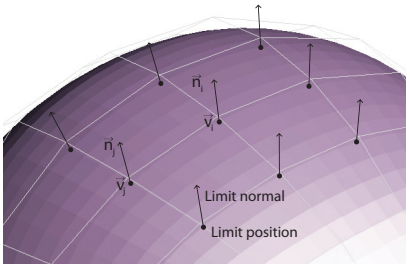


**Figure 9:** *Limit positions and normals around the one-ring neighborhood of vertex i as used in the V-spring formulation.*

The above update rule applies to the control mesh vertices associated with the network curves. Mesh vertices for the interior regions, on the other hand, are left free. To obtain smooth final surfaces, we apply a V-spring smooth-

© The Eurographics Association 2011.

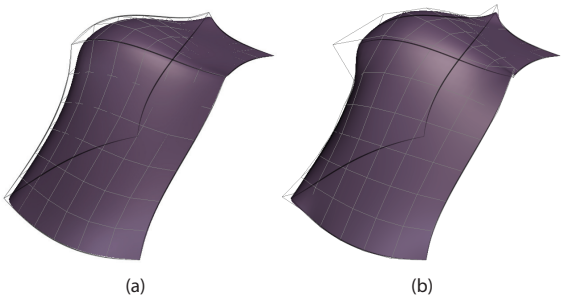


**Figure 10:** *(a) Control mesh vertices initially lie on the curve network but the limit surface lies far away. (b) The result of iterative update of the limit surface.*

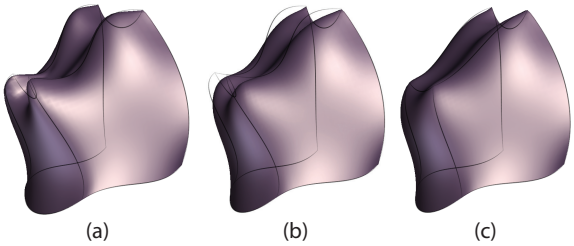


**Figure 11:** *(a) Original curve network results in artifacts in the limit surface. (b) The surface is smoothed independently from the curve network. (c) The curve network is adjusted to match the underlying surface.*

ing scheme [YFSH99] to these interior vertices. Unlike thin-plate energy minimization approaches(also used in [HKD93]), V-spring scheme is known to minimize the variation of curvature, and requires the knowledge of both the position and normal of a vertex. In our approach, we use the position and normal vectors of the limit surface vertices to compute a displacement vector:

$$\triangle \vec{v}_i^{\,Vspring} = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{\|\vec{v}_j-\vec{v}_i\|}\left[\frac{(\vec{v}_j-\vec{v}_i)\cdot(\vec{n}_j+\vec{n}_i)}{1+(\vec{n}_j\cdot\vec{n}_i))}\right]\vec{n}_j + \underbrace{\left[\triangle \vec{v}_i^{\,Laplace} - (\triangle \vec{v}_i^{\,Laplace}\cdot\vec{n}_i)\right]}_{regularization} \quad (4)$$

where $\vec{n}_i$ and $\vec{n}_j$ are the unit normal vectors of vertices $i$ and $j$ respectively. The regularization term aims to distribute the vertices uniformly through small adjustments in the tangent plane. We apply this vector to the corresponding control mesh vertex. This approximate V-spring smoothing (computing displacement from the limit surface but applying it to the control mesh) works in practice as the limit surface positions are computed as a linear combination of the control mesh vertices.

The two control mesh update rules - curve network interpolation and V-spring - are simultaneously applied. The balance between the interpolation accuracy and overall surface smoothness is controlled by $\lambda$ in Eqn. 3. Figure 10 shows the resulting control mesh and limit surface after the iterative modification. Note that the limit surface interpolates the curve network at points determined by the initial resolution of the control mesh. For curve network interpolation, a least squares formulation proposed by [HKD93] could also be used. However, our approach allows both interpolation and V-spring forces to be handled similarly, thus making the update rule simpler to implement.

#### 4.3.3. Curve Network Updating

When curve interpolation is heavily favored, the above processes may result in surfaces that exhibit undesirable artifacts. In such situations, the underlying surface can be brought to a desirable smoothness, and the network can be adjusted to conform to the surface. For this, the user selects a set of curves for adjustment. These curves are recalculated to approximate the limit positions of the associated vertices through cubic curve fitting. An example is shown Fig. 11.

## 5. Results and Discussions

We implemented the proposed method in C++ using Qt development environment for interfacing, and OpenGL for graphics. A Wacom Intuos tablet is used for sketch input. For efficient rendering of subdivision surfaces, the approximate Catmull-Clark formulation by Loop and Schaefer [LS08] is used. On a dual core laptop computer with 3 GB of RAM and NVidia Quattro graphics card, the typical operations are at interactive speeds. The examples shown throughout the paper took from 2 to 14 minutes to construct by one of the authors.

Figure 12 illustrates the typical modeling process on a telephone handset. First, a simple box is constructed by sketching symmetric curves and then surfaced (Fig. 12a). The curves are then modified to achieve the c-bend of the handset and the surface is updated to match the changes (Fig. 12b). The curve network is further developed with additional curves and simultaneously surfaced (Fig. 12c). After further additions and modifications (Fig. 12d), the surface is smoothed to a desired level (Fig. 12e). The curves are then adjusted to match the desired final surface (Fig. 12 bottom row). This example took 6 minutes to construct. Figure 13 shows other examples. The human torso consists of 75 individual curves and took 14 minutes to complete.

Currently, our approach is designed for four sided loops only. However, automatic loop identification can be trivially extended to lower or higher n-gons without any change. For initial control mesh calculation, however, we will incorporate more general techniques similar to those proposed by Kuriyama [Kur94]. Additionally, while we have designed our surfaces to be smooth everywhere, we recognize certain applications may require sharp edge generations. For

© The Eurographics Association 2011.

this, we will extend our subdivision models to incorporate the conventional crease formulations suggested for Catmull Clark surfaces.

Our surface modeling approach requires the creation of closed loops. Hence, curves that do not meet with other curves are currently not processed by our system. Note that this capability is readily supported in polygonal-based approaches such as Fibermesh [NISA07]. We seek to extend our approach to such configurations using conventional control meshes, but developing an advanced association method between the sketched curves and the control mesh.

## 6. Conclusions

In this paper, we describe a new sketch-based 3D modeling system that enables the user to rapidly design smooth surfaces using sketched curve networks. The user draws a 3D curve network which is automatically patched with smooth subdivision surfaces. The resulting surfaces can be further adjusted by modifying the underlying curve network. The curve-level interaction with the designed surfaces takes advantage of the stroke-based input. We demonstrated the system with examples exhibiting different topologies. Our current experiments have indicated that the proposed work can be a suitable approach to geometries commonly encountered in industrial product design.

## Acknowledgments

This work was supported in part by the National Science Foundation grant CAREER CMMI-0846730.

## References

[BBS08] BAE S., BALAKRISHNAN R., SINGH K.: Ilovesketch: as-natural-as-possible sketching system for creating 3d curve models. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (2008), ACM New York, NY, USA, pp. 151–160.

[BHSF09] BEIN M., HAVEMANN S., STORK A., FELLNER D.: Sketching subdivision surfaces. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling* (2009), ACM, pp. 61–68.

[CC78] CATMULL E., CLARK J.: Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 6 (1978), 350–355.

[Coo67] COONS S.: *Surfaces for computer-aided design of space forms*. Tech. rep., MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1967.

[DAJ03] DE ARAUJO B., JORGE J.: Blobmaker: Free-form modelling with variational implicit surfaces. In *Proceedings of* (2003), vol. 12, Citeseer.

[DDGG05] DAS K., DIAZ-GUTIERREZ P., GOPI M.: Sketching free-form surfaces using network of curves. In *Proceedings of eurographics workshop on sketch-based interfaces and modeling (SBIMŠ05)* (2005), Citeseer.

[ES07] EISSEN K., STEUR R.: *Sketching: drawing techniques for product designers*. Page One, 2007.

[HKD93] HALSTEAD M., KASS M., DEROSE T.: Efficient, fair interpolation using catmull-clark surfaces. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (1993), ACM, pp. 35–44.

[HPS10] HERMANN T., PETERS J., STROTMAN T.: Constraints on curve networks suitable for g 2 interpolation. *Advances in Geometric Modeling and Processing* (2010), 77–87.

[JS11] JORGE J., SAMAVATI F.: *Sketch-based interfaces and modeling*. Springer, 2011.

[KDS06] KARA L. B., D'ERAMO C. M., SHIMADA K.: Pen-based styling design of 3d geometry using concept sketches and template models. In *SPM '06: Proceedings of the 2006 ACM symposium on Solid and physical modeling* (New York, NY, USA, 2006), ACM, pp. 149–160.

[KH06] KARPENKO O. A., HUGHES J. F.: Smoothsketch: 3d free-form shapes from complex sketches. *ACM Transactions on Graphics 25/3* (2006), 589–598.

[KS07] KARA L. B., SHIMADA K.: Sketch-based 3d-shape creation for industrial styling design. *IEEE Computer Graphics and Applications 27*, 1 (2007), 60–71.

[Kur94] KURIYAMA S.: Surface modelling with an irregular network of curves via sweeping and blending. *Computer-Aided Design 26*, 8 (1994), 597–606.

[Lev99] LEVIN A.: Interpolating nets of curves by smooth subdivision surfaces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), ACM Press/Addison-Wesley Publishing Co., pp. 57–64.

[LS08] LOOP C., SCHAEFER S.: Approximating catmull-clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics (TOG) 27*, 1 (2008), 8.

[MS92] MORETON H., SÉQUIN C.: Functional optimization for fair surface design. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (1992), ACM, pp. 167–176.

[NISA07] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: Fibermesh: Designing freeform surfaces with 3d curves. In *ACM SIGGRAPH 2007* (San Diego, USA, 2007), ACM Transactions on Computer Graphics.

[NKS09] NASRI A., KARAM W., SAMAVATI F.: Sketch-based subdivision models. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling* (2009), ACM, pp. 53–60.

[NSACO07] NEALEN A., SORKINE O., ALEXA M., COHEN-OR D.: A sketch-based interface for detail-preserving mesh editing. In *ACM SIGGRAPH 2007* (2007), ACM, pp. 42–es.

[OSSJ09] OLSEN L., SAMAVATI F. F., SOUSA M. C., JORGE J. A.: Sketch-based modeling: A survey. *Computers & Graphics 33*, 1 (2009), 85 – 103.

[SKSK09] SCHMIDT R., KHAN A., SINGH K., KURTENBACH G.: Analytic drawing of 3d scaffolds. In *ACM SIGGRAPH Asia 2009 papers* (2009), ACM, pp. 1–10.

[SL96] SHPITALNI M., LIPSON H.: Identification of faces in a 2d line drawing projection of a wireframe object. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 18*, 10 (1996), 1000–1012.

[SWSJ06] SCHMIDT R., WYVILL B., SOUSA M. C., JORGE J. A.: Shapeshop: sketch-based solid modeling with blobtrees. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses* (New York, NY, USA, 2006), ACM, p. 14.

[SWZ04] SCHAEFER S., WARREN J., ZORIN D.: Lofting curve networks using subdivision surfaces. In *Proceedings of the 2004*

© The Eurographics Association 2011.

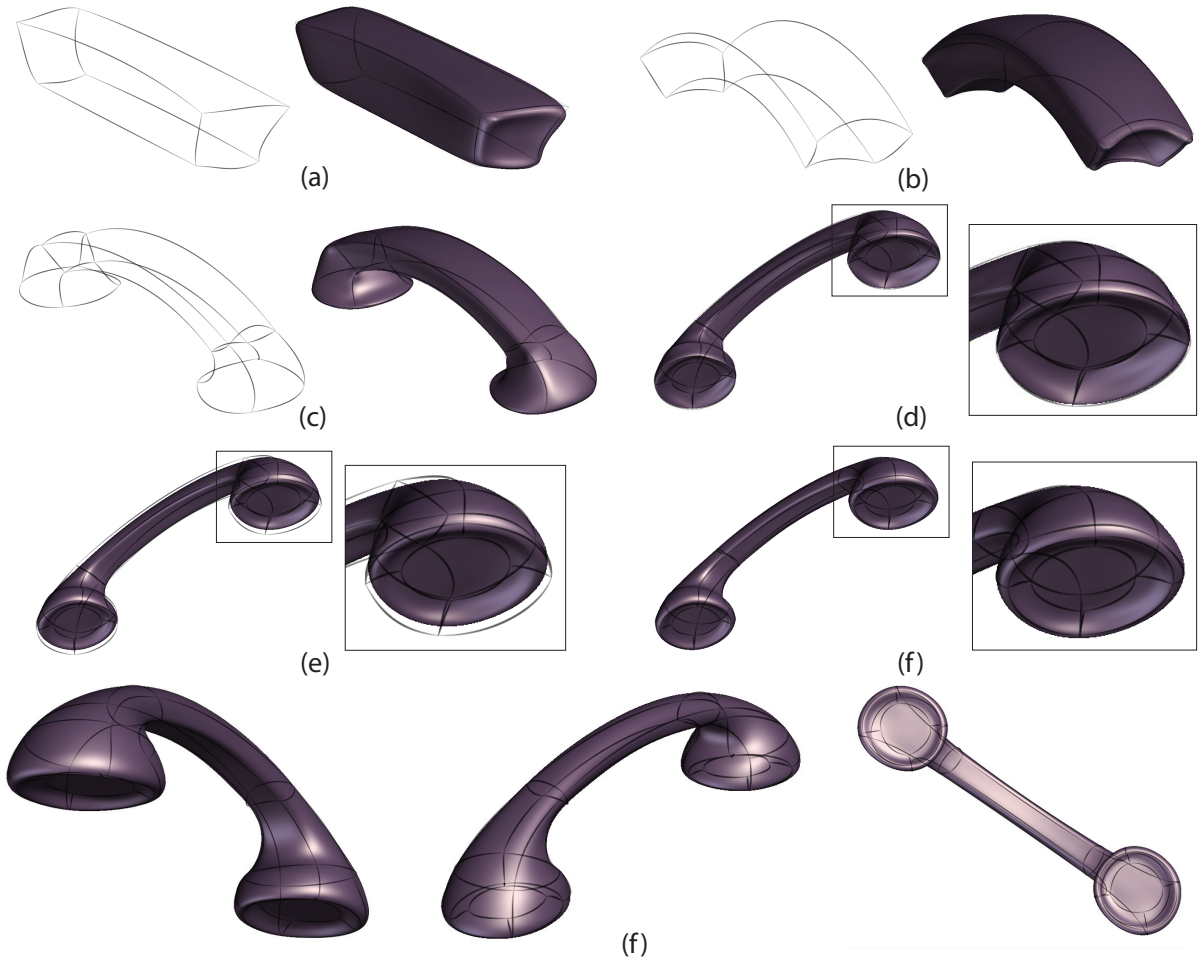


**Figure 12:** *Construction of a phone handset.*

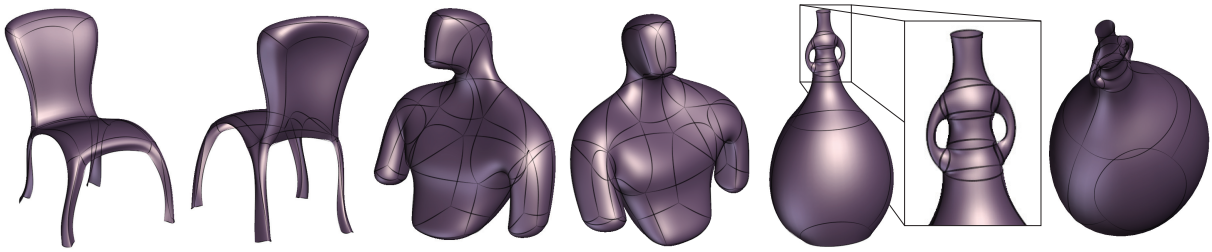

**Figure 13:** *Various models created using our system.*

*Eurographics/ACM SIGGRAPH symposium on Geometry processing* (2004), ACM, pp. 103–114.

[TZF04] TAI C., ZHANG H., FONG J.: Prototype modeling from sketched silhouettes based on convolution surfaces. In *Computer Graphics Forum* (2004), vol. 23, Wiley Online Library, pp. 71–83.

[WW94] WELCH W., WITKIN A.: Free-form shape design using triangulated surfaces. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (1994), ACM, pp. 247–256.

[YFSH99] YAMADA A., FURUHATA T., SHIMADA K., HOU K.: A discrete spring model for generating fair curves and surfaces. *pg* (1999), 270.

© The Eurographics Association 2011.