Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2011
August 28-31, 2011, Washington, DC, USA

DETC2011-48414

APPROXIMATE SURFACING OF CURVE CLOUDS FOR CONCEPTUAL SHAPE CREATION AND EVALUATION

Erhan Batuhan Arisoy* **Gunay Orbay†** **Levent Burak Kara‡**
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

ABSTRACT

In product design, designers often create a multitude of concept sketches as part of the ideation process. Transforming such sketches to 3D digital models usually require special expertise and effort due to a lack of suitable Computer Aided Design (CAD) tools. Although recent advances in sketch-based user interfaces and immersive environments (such as augmented/virtual reality) have introduced novel curve design tools, rapid surfacing of such data remains an open challenge. To this end, we propose a new method that enables a quick construction of approximate surfaces from a cloud of 3D curves that need not be connected to one another. Our method first calculates a vector field by discretizing the space in which the curve cloud appears into a voxel image. This vector field drives a deformable surface onto the 3D curve cloud thus producing a closed surface. The surface smoothness is achieved through a set of surface smoothing and subdivision operations. Our studies show that the proposed technique can be particularly useful for early visualization and assessment of design ideas.

INTRODUCTION

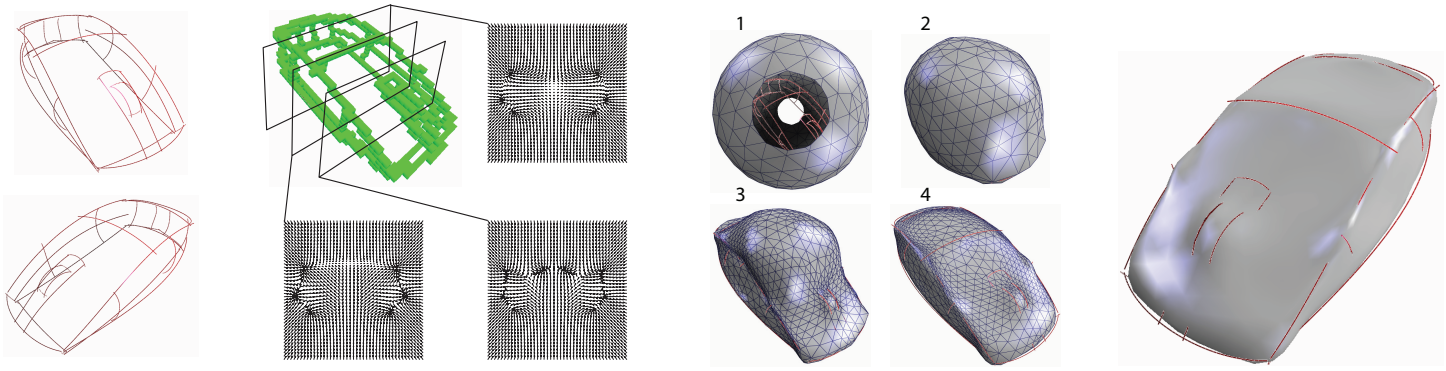
Advances in 3D product form design have resulted in a large number of sophisticated computer software for a range of different geometric modeling applications. However, many of these tools require substantial experience and specialization in the underlying representations and associated geometric operations. This results in many designers to still utilize conventional media such as thumbnail sketches for idea generation and exploration, and defer 3D computer modeling only after the ideas have sufficiently matured. As a result, only a small subset of the generated ideas may be considered for further stages while many promising ones are abandoned prematurely.

To alleviate these difficulties, recent studies have focused on direct 3D modeling techniques. A body of work has focused on 3D geometry creation methods utilizing sketch-based user interfaces [1–4], and systems deployed in immersive Virtual Reality (VR) environments [5, 6]. Although these studies have made advances in better utilizing 2D and 3D input devices in curve and surface design, they still require a careful dictation and control of the geometric data without providing the full conceptual freedom of a paper sketching interface.

In this work, we propose a new method that takes as input roughly sketched 3D curves created through a sketch-based user interface or through an immersive design environment, and produces an approximate closed surface that matches the curve cloud with a user controllable smoothness. The key advance in this work is its ability to operate on curve clouds which may not necessarily form a network topology. Hence, the main challenge in the proposed work is to identify a surface that closely interpolates and approximates the constituent curves, while preserving a desired level of local smoothness. The proposed method has the following specific contributions:

1. Generating approximate closed surfaces without holes from curve clouds with arbitrary topology

*earisoy@andrew.cmu.edu
†gorbay@andrew.cmu.edu
‡lkara@andrew.cmu.edu Address all correspondence to this author.



(a) The input is a cloud of 3D curves

(b) The discretized 3D voxel image of the sketch and the calculated guidance field

(c) An elastic surface is gradually deformed to approximate the ingredient 3D curves

(d) The final approximate surface after surface fairing operations

FIGURE 2. A breakdown of our approach: (a) a curve cloud is input to the system, (b) the guidance vector field is calculated within the domain, (c) the initial surface is deformed toward the input curves, (d) the final surface is achieved after surface smoothing and subdivision operations.

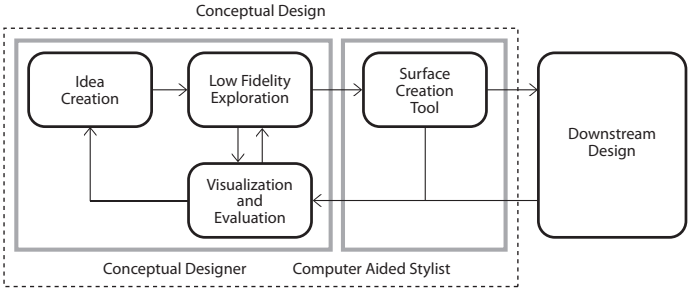


FIGURE 1. Current ideation process in conceptual design. We provide a new approach in surface creation tools which allow quick visualization of conceptual ideas.

2. Preserving local smoothness while approximating the curve cloud
3. Enabling early and quick visualization of the curve cloud at early design stages for rapid evaluation and refinement
4. Bridging the gap in the design process by providing a means to automatically transform designers' raw, unrefined shape ideas into geometric data suitable for further beautification and refinement using conventional CAD tools

In a typical scenario, our method takes as input a set of 3D curves (Fig. 2.a) where each stroke is considered as a single curve. Next, a guidance vector field is calculated by quantizing the curve set into a binary voxel image (Fig. 2.b). Once created, the guidance field helps gradually deform an elastic surface instantiated in the domain until the surface captures the object shape suggested by the curve cloud (Fig. 2.c). Once the initial working surface is obtained, the surface can be further refined through conventional mesh operations as desired (Fig. 2.d). The user is able to control both the surface deformation process and the post surface fairing operations while interactively observing the modified surface. In the following sections, we review the literature relevant to our work, followed by a technical description of the proposed method. We demonstrate the utility of the proposed method with example cases.

RELATED WORK

Developing CAD tools for early design phases has been an active area of research for product form design. Recent studies have explored the use of informal inputs such as sketch-based solid modeling. Most of these studies aim to provide tools for rapid visualization and evaluation of product forms. These studies can be categorized into two main groups. The first group focuses on user-guided surface modeling techniques where the designer progressively creates an intended geometry through a sketch-based interface [1, 2, 7–9]. Usually, the underlying mathematics of these operations and surface definitions are hidden from the user. It is argued that these techniques are more suitable for inexperienced users due to the simplicity of the interactions [2]. Many of these techniques require each constructed curve to be closely approximated by the fitted surfaces, thus making the design of each curve a delicate and binding process.

Nealen *et al.* [2] presented such a system called Fibermesh for creating free form surfaces. The user first draws a simple closed stroke in 2D and the system automatically generates a shape whose contour matches the user drawn 2D closed curve via inflation. Various operators such as sketching, pulling and pushing allow the designer to modify the initial surface. Earlier works from Igarashi such as Teddy [1] and Karpenko's system [8] are similar in spirit. Likewise, Siguhara *et al.* [4] proposed a free form deformation technique suitable for implicit surface representations where the designer creates intended shapes using interactive push and pull operations. Our approach aims to provide a modeling experience more commensurate with industrial mod-

eling in that it allows the designer to layout a rough wireframe model that can be surfaced when desired, without prematurely exposing the user to a surface model that needs to be successively modified.

Second group of surface modeling techniques consists of automatic methods that produce free form surfaces. These studies are based on different concepts such as optimization [2,10,11], partial differential equations (PDE) [12–14], parametric patches [15], subdivision surfaces [16], potential field information [4,17,18]. While creating high quality surfaces, these approaches require the user to understand and apply the boundary conditions for the desired outcomes. Such specifications, however, are typically central in the detailed design stages and may be a hindrance in the early stages of the design.

Our work is motivated by the idea that a product design form can be descibed using a sparse collection of 3D curve clouds. The notion of using a complete set of curve network to generate an interpolating 3D surface which closely satisfies these curves has been employed by traditional CAD tools for a long time and our aim in this paper is providing a modeling experience more commensurate with industrial modeling in that it allows the designer to layout a rough wireframe model that can be surfaced when desired, without prematurely exposing the user to a surface model that needs to be successively modified. The main advantage of our proposed algorithm is that it does not require a fully connected set of curves as input to generate a free form surface. In other words, it relievies many topological and geometric restrictions of a classic curve network which allows us to handle a broad spectrum of curve networks for exploration and ideation phases at early conceptual design stages. Moreover, our algorithm with select and modify tools combines surface generation task with sketch-based mesh editting operations which provides local differential surface control too.

Xu *et al.* [19] proposed a surfacing approach involving medical image segmentation. The algorithm is based on the calculation of a vector field which is solved as the minimum of an energy functional which drives initialized contours toward object boundaries. We adopt the same idea presented in this work of creating a vector field for deforming an elastic object. However, the main difficulty in our approach is that the input geometric data representing an object is often defined only partially and approximately along its boundaries, while no information exists to reconstruct the surfaces spanning these boundaries. Our approach aims to overcome this difficulty by delegating the internal energy of the elastic deformable surface to shape the missing pieces of the geometry in a way that is plausible to the designer.

TECHNICAL DETAILS

Overview

In this paper, we propose a new surfacing method that aims to bring CAD support to the early stages of product form design which will enable designers to quickly visualize and evaluate their emerging ideas. Our approach takes as input 3D curve clouds without any specific topology requirements, and generates a closed surface that approximates the constituent curves while minimizing a prescribed smoothness criterion. Our approach operates in two main steps: (1) Guidance vector field calculation, and (2) Deformable surface initialization and deformation. In the first step, we calculate a discrete guidance vector field from the curves making up the wireframe model. To do so, we use the Gradient Vector Flow (GVF) Field [19] which is a method primarily used for medical image segmentation. We start by discretizing the domain and initiating a gradient vector field using the seed curves, followed by a calculation of the flow field through diffusion of the gradient field. In the next step, we initialize a genus 0 surface (*i.e.* closed surface without through-holes) which initially embodies the curves, and deform it under forces emanating from the guidance field vectors and the smoothness preserving criterion respectively. Initialized seed surface for deformation could be a higher genus surface and final surface after deformations and surface fairing operations would be homemorphic to this seed surface which is a genus 0 sphere in this case.

Generation of Input Curve Clouds

The curve clouds that are taken as input can be generated through a number of different methods such as through 3D input devices in augmented reality environments [5,6], or through existing 3D wireframe modeling approaches that may utilize sketch-based user interfaces [3,4]. In this work, the curve clouds we use are generated using a 2D sketching interface capable of calculating 3D locations of symmetric curves from the projections of the symmetric pair on screen coordinates like in the work ILoveSketch [20]. One reason for this choice is that sketch-based user interfaces are more accessible and familiar to use in comparison to virtual reality systems. However, the proposed method is not limited to the curve models created using this interface.

In our approach, users sketch pairs of symmetric curves on the drawing surface. Figure 3 illustrates the idea. Each curve is first converted to a cubic Bezier curve with four control points in the image plane. The 3D configuration of the symmetric curve pairs is found using least squares minimization approach. Given the symmetry plane defined by a position vector S, and a normal vector N, we first construct two rays, $\vec{w}_A$ and $\vec{w}_B$, emanating from the viewpoint $\vec{C}$ toward the symmetric pair $\vec{P}_A$ and $\vec{P}_B$, respectively. We write the position vectors $\vec{P}_A$ and $\vec{P}_B$ with respect to $\vec{C}$ and the symmetry plane. Vector algebra enables the calculation of $\vec{P}_A$ and $\vec{P}_B$ through the least squares solution of the following matrix:

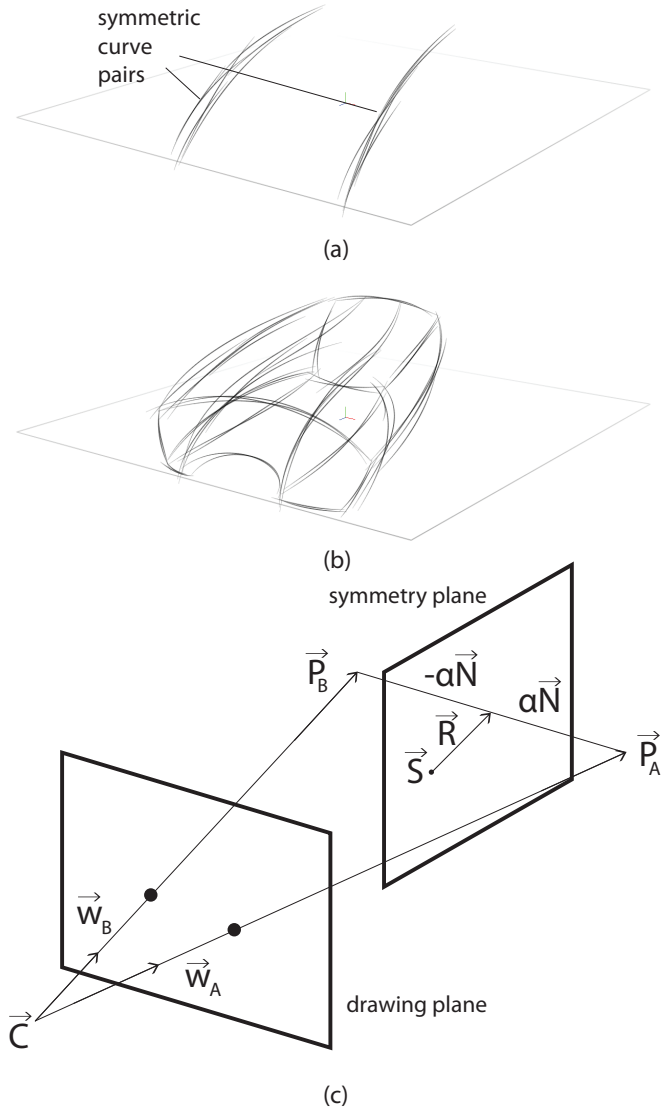


FIGURE 3. Calculating the 3D positions of point pairs symmetric about a symmetry plane from their projections on the viewing plane.

$$\begin{bmatrix} w_{Ax} & -w_{Bx} & 2N_x \\ w_{Ay} & -w_{By} & 2N_y \\ w_{Az} & -w_{Bz} & 2N_z \\ \vec{w}_A \cdot \vec{N} & -w_{Bx} & 0 \end{bmatrix} \begin{Bmatrix} d_A \\ d_B \\ \alpha \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 2\vec{S} \cdot \vec{N} - 2\vec{C} \cdot \vec{N} \end{Bmatrix} \quad (1)$$

where d_A and d_B are the distances on rays $\vec{w}_A$ and $\vec{w}_B$ respectively between $\vec{C}$ and $\vec{P}_A$, $\vec{P}_B$, and α is the perpendicular distance from the symmetry plane to $\vec{P}_A$ and $\vec{P}_B$. We solve the above least squares problem for all symmetric control point pairs to determine the position and orientation of the curve pairs in 3D.

The user creates as many such symmetric curve pairs as desired. The output of this process is a curve cloud containing the raw curves that are not necessarily connected to one another. As such, this kind of geometric content is not suitable for surfacing using conventional methods that require connected wireframe models [1, 2].

Calculation of the Guidance Field

Requirements of the Guidance Field To generate approximate closed surfaces without holes from curve clouds, we aim to compute a guidance vector field which will deform an elastic surface toward the input curves. In order to determine such a vector field, we identify three requirements that the field should satisfy.

The first requirement is the existence of the field (*i.e.* be nonzero) everywhere in the input domain. This feature ensures that a surface instantiated far away from the curve cloud can be appropriately driven toward the cloud. Conventional techniques using gradient information only [21, 22] fail to satisfy this requirement as image gradients quickly diminish away from the object. The second requirement is that the vector field should generate vectors that can spear through concave regions without diminishing at the entry of the concavity. Conventional gradient vectors often fail to reproduce such regions as shown in Fig. 4. Finally, an appropriate field must direct toward the 3D curves in the vicinity of the curves. Although the last requirement is straightforward via gradient calculations only, it is not trivial to establish a vector field that simultaneously satisfies all three requirements.

To this end, we employ the gradient vector flow field (GVF), a technique for 3D segmentation of medical images [19]. The key observation underlying GVF is that the above requirements can be obtained by diffusing the initial gradient vector field obtained from the raw image. This approach is conceptually analogous to solving the heat diffusion equation to determine the steady state temperature field in a domain with adiabatic boundaries and an object with heat generation dependent to its temperature. However, the main difference is that in GVF, the diffusion of a vector field is computed rather than that of a scalar field.

A primary difficulty in our problem is that the input curve networks are usually specified along the boundaries of the intended object. The middle of the surfaces, however, are generally devoid of such curves unless the user dictates details in those regions. As a result, unlike the case in segmentation of medical image, our approach requires a careful negotiation between the internal energy of the deformable surface and the external force field that aims to deform it. This balance is necessary to generate surfaces that capture large regions of surfaces in ways plausible to the user, when there is no data to support their particular configurations. The following sections detail these effects and describes a final vertex-based deformation procedure taking into account this phenomena.

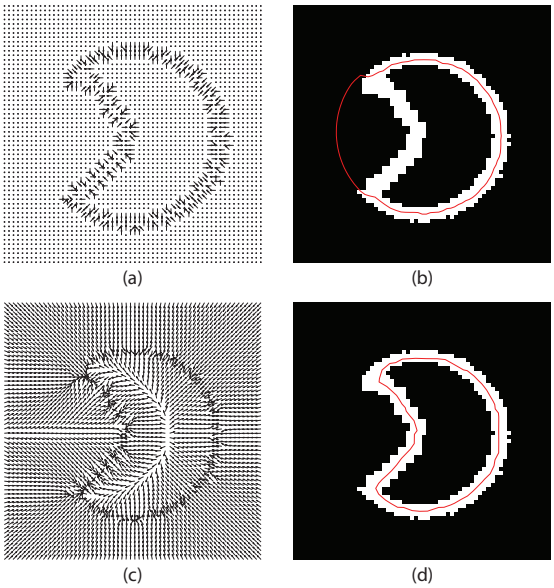


FIGURE 4. (a) The comparison of a gradient vector field to (c) the Gradient Vector Flow (GVF) field. (b) The active contours cannot penetrate into the concave regions with the simple gradient field (d) whereas it can with the GVF field.

Calculation of the Guidance Field GVF is the continuous 3D vector field $\vec{V}(x,y,z) = [u(x,y,z), v(x,y,z), k(x,y,z)]$ that minimizes the following energy functional [19]:

$$\varepsilon = \iiint \underbrace{\mu(u_x^2+u_y^2+u_z^2+v_x^2+v_y^2+v_z^2+k_x^2+k_y^2+k_z^2)}_{\text{Smoothness}} + \underbrace{\|\nabla f\|^2\|\vec{V}-\nabla f\|^2}_{\text{Pointing to objects}} dxdydz \tag{2}$$

where $\vec{f}(x,y,z)$ is a 3D voxel image, ∇ is the gradient operator in Cartesian coordinates, $\vec{u}(x,y,z)$, $\vec{v}(x,y,z)$ and $\vec{k}(x,y,z)$ are the x, y and z components of the vector field at the point (x,y,z), and the subscripts denote partial derivatives. μ is a coefficient which controls the magnitude of diffusion. In this functional, the first term aims to preserve the smoothness of the vector field. The second term aims to make the vector field equal to the image gradient when the solution is near the curve cloud. In the vicinity of the curve cloud, the second term dominates as the gradient vector's magnitude will be large compared to the other terms. Hence, the vector field will aim to match the gradient vector to minimize the functional. Far away from the curve cloud, however, gradient vectors will vanish thus making the first term in the functional to dominate. This, in turn, is minimized by minimizing the variation in the vector field. The solution field $\vec{V}(x,y,z)$ minimizing this energy functional can be calculated using calculus of variation on three decoupled Euler equations:

$$\begin{aligned}
\mu\nabla^2 u-(u-f_x)(f_x^2+f_y^2+f_z^2)=0\\
\mu\nabla^2 v-(v-f_y)(f_x^2+f_y^2+f_z^2)=0\\
\mu\nabla^2 k-(k-f_z)(f_x^2+f_y^2+f_z^2)=0
\end{aligned} \tag{3}$$

where the Laplacian operator is defined as:

$$\nabla^2 u(x,y,z) = \frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2} \tag{4}$$

The solutions to the Euler equations $\vec{u}(x,y,z)$, $\vec{v}(x,y,z)$, $\vec{k}(x,y,z)$ are the vector field components in x, y, and z directions and can be solved independently. The diffusion coefficient can be adjusted based on the desired diffusion level. In our approach, it is kept constant at 1 for the solution in each direction. u, v and k are initialized as the gradient of the 3D image in the directions x, y, and z respectively. We solve these equations by discretizing the domain as explained next.

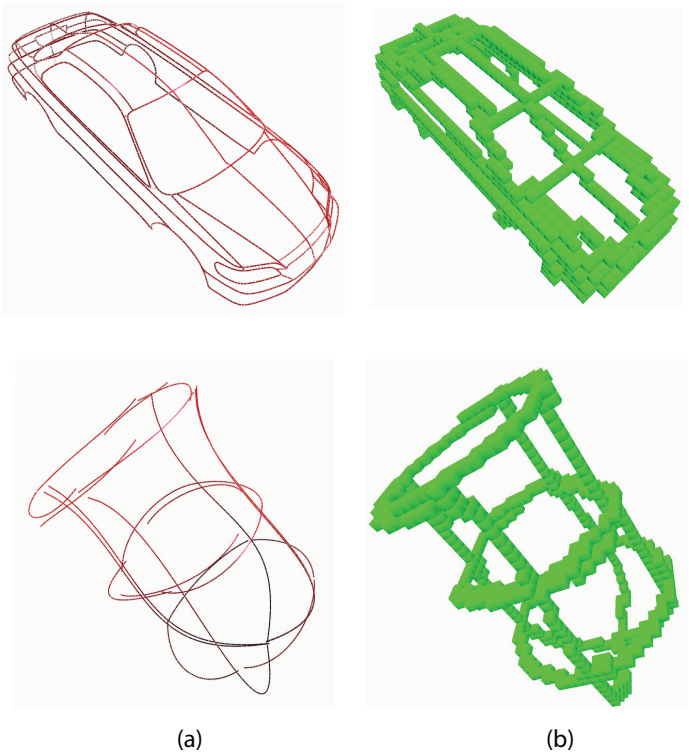


FIGURE 5. The space containing the input curves in (a) are discretized into binary voxel images in (b).

Discretization of the Continuous Domain and Equations The 3D curve clouds are represented as a collection of polylines in space and are discretized according to a user specified resolution. To begin, we first transform the domain by mapping the curve cloud into a unit cube, and the cube is uniformly expanded 30% in all directions. The domain is then discretized into an $N \times N \times N$ grid. Next, the curve cloud is quantized as a binary 3D voxel image in the grid. Figure 5 illustrates the idea.

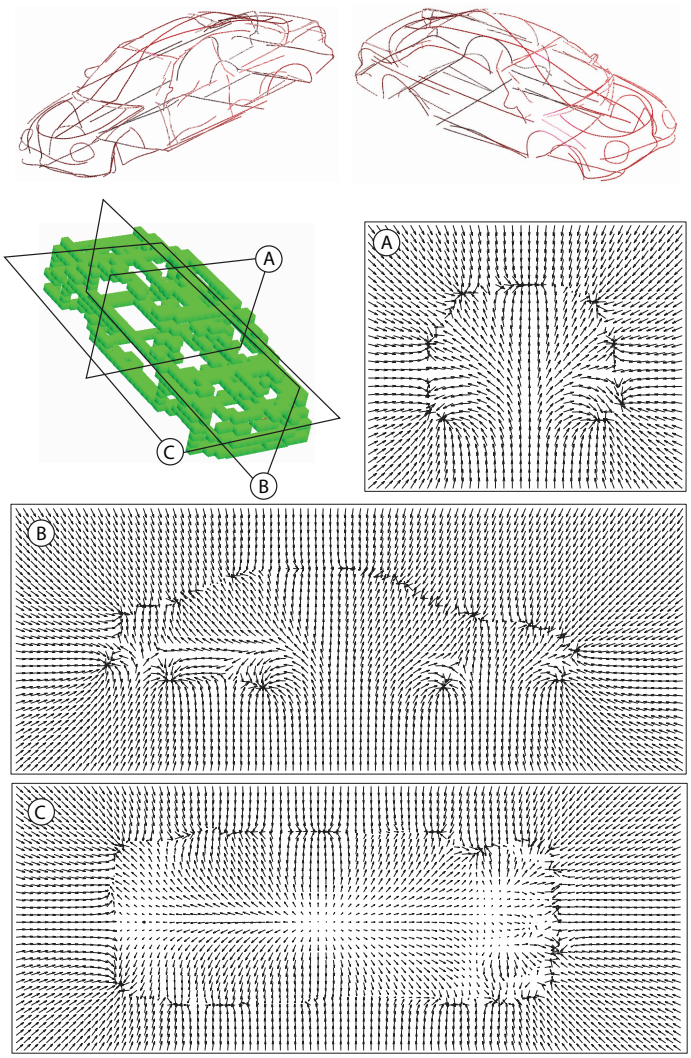


FIGURE 6. Gradient vector flow field of a car model on the specified cross sections.

In the discretized domain, we approximate the three Euler equations through second order finite difference formulations. The following equations illustrate the approximation of the second order derivatives [23]:

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{(m,n,z)} = \frac{u_{m+1,n,z} + u_{m-1,n,z} - 2u_{m,n,z}}{\triangle x^2}$$
$$\left.\frac{\partial^2 u}{\partial y^2}\right|_{(m,n,z)} = \frac{u_{m,n+1,z} + u_{m,n-1,z} - 2u_{m,n,z}}{\triangle y^2} \quad (5)$$
$$\left.\frac{\partial^2 u}{\partial z^2}\right|_{(m,n,z)} = \frac{u_{m,n,z+1} + u_{m,n,z-1} - 2u_{m,n,z}}{\triangle z^2}$$

The Laplace operator has to be discretized for each cube with indices (m, n, z) in our 3D voxel image using the previous second order central difference schemes shown in Eqn. [5]:

$$\begin{aligned}\nabla^2 u(x,y,z)\big|_{(m,n,z)} = &\frac{u_{m+1,n,z} + u_{m-1,n,z} - 2u_{m,n,z}}{\triangle x^2} \\ &+ \frac{u_{m,n+1,z} + u_{m,n-1,z} - 2u_{m,n,z}}{\triangle y^2} \\ &+ \frac{u_{m,n,z+1} + u_{m,n,z-1} - 2u_{m,n,z}}{\triangle z^2}\end{aligned} \quad (6)$$

The solution of the above equations is facilitated by initializing the solution field u, v and k to be the gradient of the quantized curve cloud. The equations are solved iteratively until the difference between successive iterations becomes less than a prescribed threshold.

Figure 6 shows the final vector fields on the specified cross sections of a car model quantized into a $91 \times 91 \times 91$ grid.

Surface Initialization and Deformation After the calculation of the vector flow field, a sphere represented as a triangular mesh that encapsulates the constituent curves is initiated on the voxel image. The vertices of this triangular mesh is then iteratively deformed inside the vector field until the surface attains a stable configuration around the curve cloud. In each iteration, the mesh vertices experience external deformation forces from the vector field, as well as internal forces that aim to preserve the smoothness of the mesh during the iterations. Each iteration requires the computation of such external and internal forces at arbitrary positions in the domain. For the external field forces, we use a tri-linear interpolation of the discretized GVF field that helps approximate the vector field at arbitrary points in the domain. The internal forces, on the other hand, are computed directly from the mesh geometry and thus do not require interpolation from the discretized grid. The following paragraphs explain the internal forces applied to the deformable mesh and Fig. 7 illustrates the overall surface development algorithm.

Internal forces: These forces work to minimize the stretch energy of the deformable mesh while maintaining a uniform spacing along the mesh. For each vertex, these forces are computed

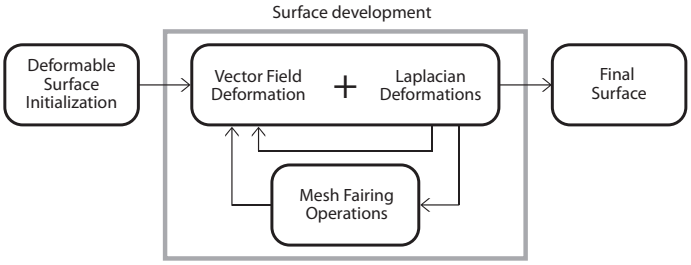


FIGURE 7. The surface deformation cycle of the proposed approach. The surface is deformed according to the vector field while the surface smoothness is maintained through surface fairing and subdivision operations.

from the one-ring neighborhood as shown in Fig. 8. The first force uses the Laplacian operator defined on a discrete mesh as follows:

$$\triangle \vec{v}_i^{\,Laplace} = \left(\frac{1}{n} \sum_{j=1}^{n} \vec{v}_j \right) - \vec{v}_i \tag{7}$$

where $\triangle \vec{v}_i^{\,Laplace}$ is the Laplacian displacement, n is the number of vertices in the one-ring neighborhood of vertex i, $\vec{v}_i$ is its position vector, and $\vec{v}_j$ is the position vector of the j^{th} neighbor (Fig. 8.a). While a powerful scheme, this force smooths the mesh via local flattening at the expense of reducing the volume. The second internal force is based on the Biharmonic operator as shown in Fig. 8.b. The Biharmonic forces acting on a discrete mesh is calculated as follows:

$$\triangle \vec{v}_i^{\,Biharmonic} = \left(\frac{1}{n} \sum_{j=1}^{n} \triangle \vec{v}_j^{\,Laplace} \right) - \triangle \vec{v}_i^{\,Laplace} \tag{8}$$

The Biharmonic forces, similar to Laplacian forces, gradually smooth the mesh while regularizing the spacing throughout the mesh.

Deformation rule: The following equation sets illustrate our general deformation rule for each vertex on the elastic surface. The first equation represents the discretized form of the deformation of a dynamic elastic model in time [24] and the second equation describes the combination of internal forces with the external forces. The coefficients α_1 and α_2 are obtained empirically as 0.7 and 0.3 using different curve clouds. These terms enhance local smoothness and Fig. 9 illustrates each deformation component.

$$\triangle \vec{v}_i^{\,Deform} = \triangle \vec{v}_i^{\,GVF} + \alpha_1 \triangle \vec{v}_i^{\,Laplace} + \alpha_2 \triangle \vec{v}_i^{\,Biharmonic} \tag{9}$$

$$\triangle \vec{v}_i^{\,Total} = \omega_{\,Deform}(\triangle \vec{v}_i^{\,Deform}) + \omega_{\,Laplace}(\triangle \vec{v}_i^{\,Laplace}) \tag{10}$$

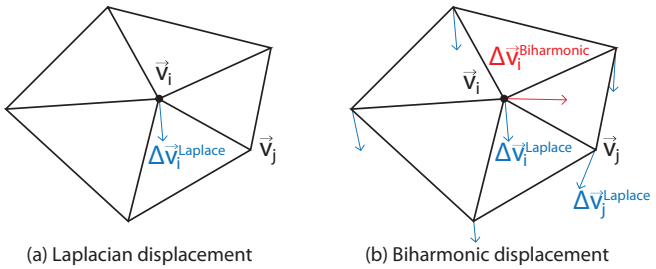


FIGURE 8. (a) The Laplacian and (b) the Biharmonic operators.

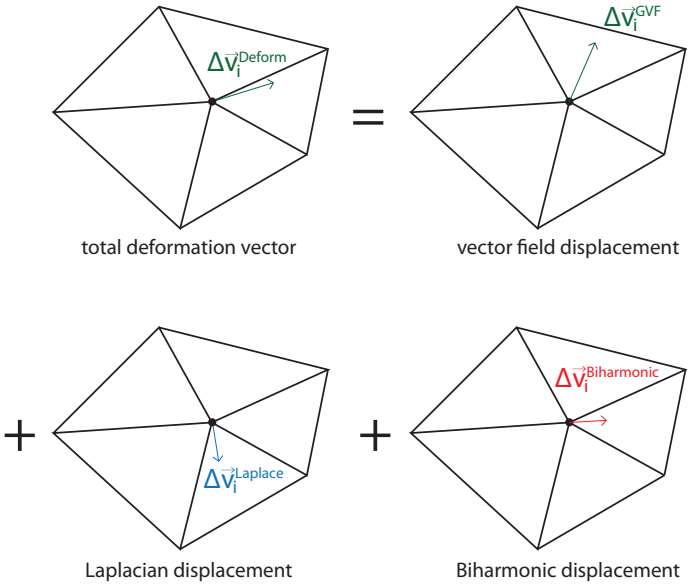


FIGURE 9. (a) The total deformation vector is the weighted sum of the displacement due to the laplacian forces, biharmonic forces and the guidance field.

The internal Laplace and Biharmonic forces work to maintain the local smoothness and the distribution of the mesh spacing while the guidance field pushes the mesh vertices toward the 3D curves. However, a simple summation of the external and internal forces causes the vertices of the deformable surface to be accumulated along the curves as the magnitude of the field forces are higher in the vicinity of the curves. This results in triangles with high aspect ratios (*i.e.* low quality) on the deformable mesh, which negatively impacts the vertex operations in the subsequent iterations. To address this problem, we use a weighted combination scheme in which weights ω_{Deform} and $\omega_{Laplace}$ are determined dynamically based on the vertices' distances to the curves during each iteration. Note that as shown in Fig. 10, ω_{Deform} favors GVF forces in the vicinity of the curves, while $\omega_{Laplace}$ favors the internal forces away from the curves. This scheme al-

lows the surface to closely approximate the curve cloud, while the internal forces promote smoothness far from the curves.

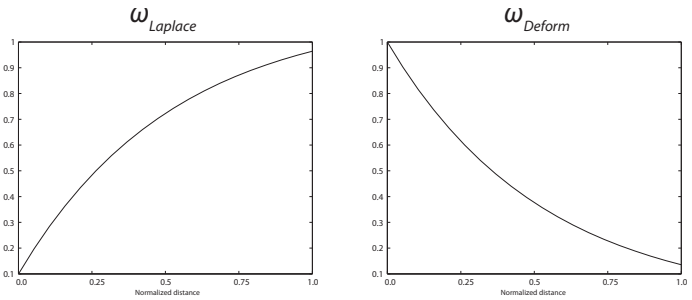


FIGURE 10. A distance map is used to adjust the relative weights of vector field displacement and Laplacian smoothing displacement.

Surface fairing operations: In addition to the above iterative deformation algorithm, our approach provides a set of user controlled surface fairing operations to be used for further surface refinement. These operations can be used in conjunction with or separately from the main deformation procedure. These operations include V-spring smoothing, Loop subdivision smoothing, Primal Triangle Quadrisection (PTQ) subdivision, and remeshing.

V-spring smoothing: This operator aims to minimize the variation of the curvature of the deformable mesh [25]. A spring is attached to each vertex in which the initial spring length represents the local curvature at that vertex. When set free, the spring set minimizes its energy by forcing neighboring vertices onto a local sphere:

$$\triangle \vec{v}_i^{\,Vspring} = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{\|\vec{v}_j-\vec{v}_i\|}\left[\frac{(\vec{v}_j-\vec{v}_i)\cdot(\vec{n}_j+\vec{n}_i)}{1+(\vec{n}_j\cdot\vec{n}_i))}\right]\vec{n}_j + \underbrace{\left[\triangle \vec{v}_i^{\,Laplace}-(\triangle \vec{v}_i^{\,Laplace}\cdot\vec{n}_i)\right]}_{\text{regularization}} \tag{11}$$

where $\vec{n}_i$ and $\vec{n}_j$ are the unit normal vectors of vertices i and j respectively. Figure 11 illustrates the calculation of this force from the one-ring neighborhood.

Loop subdivision masking: This operator makes use of the Loop subdivision scheme [26] vertex masks without any subdivisions. The displacement for each vertex is calculated using its one-ring neighborhood as follows:

$$\triangle \vec{v}_i^{\,Loop} = -\beta\vec{v}_i + \frac{1}{n}\sum_{j=1}^{n}\beta\vec{v}_j \tag{12}$$

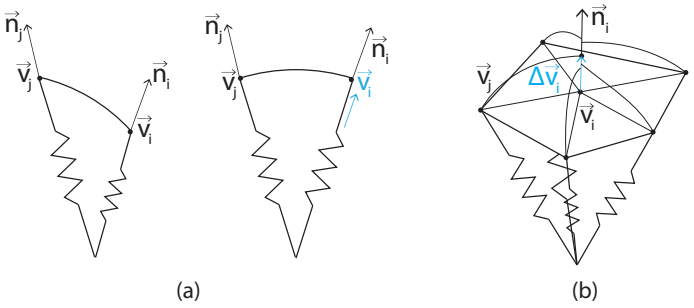


FIGURE 11. V-spring smoothing gradually minimizes the variation of curvature in the surface.

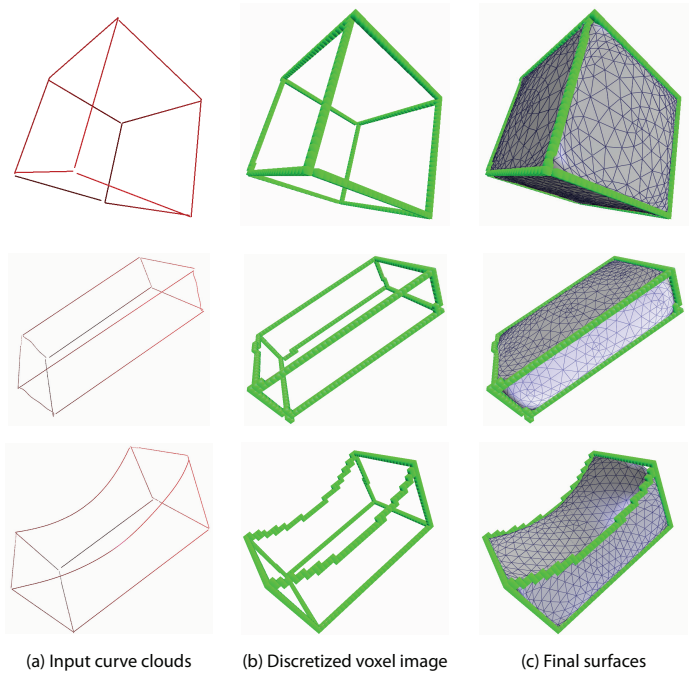


FIGURE 12. Application of the proposed algorithm on simple geometries.

where $\beta = 5/8 + (3+2\cos(2\pi/n))^2/64$.

This scheme is conceptually similar to the Laplacian forces except it provides closer control over the vertex displacements by making the displacements sensitive to the number of surrounding neighbors.

Results and Discussions

We demonstrate the utility of the proposed method on a series of different curve sets generated in a sketch-based system using a graphics tablet interface. In all cases the domain is quantized into a $91 \times 91 \times 91$ uniform grid. As shown in Fig. 12, the proposed method is capable of generating surfaces on simple ge-

ometries such as a cube and a rectangular prism. Figure 13 illustrates product form design examples, including two car bodies, a gamepad and a car seat where each example is created under 60 seconds with 6146 vertices on a 2GHz machine. The grid resolution for the domain quantization can be adjusted according to the size of the bounding box of the input curve cloud. The resulting surfaces globally approximate the input curve clouds. However, as shown, a number of local and complex details may be lost during this process in an attempt to maintain the global smoothness.For instance, the surface generated for the concept car does not reproduce the sideview mirrors which appear as smaller details compared to the rest of the body. However, we believe this effect is advantageous as it suppresses many of the undesirable artifacts that arise during the construction of the curve cloud. Moreover, resulting surfaces can be easily modified if some strokes were added to the input sketch through updating guidance field for deformations. To verify our method on ground truth data, we also applied the method to well-connected curve networks (Fig.13.c). An important observation is that, with such input, the method produces surfaces that align well with the surfaces implied by the curve network.

Conclusions and Future Work

In this paper, we propose a new method for fitting approximate surfaces to curve clouds for conceptual shape design and exploration. The proposed method enables designers to quickly construct approximate surfaces from their informal wireframe sketches. Our method is composed of two main steps; calculation of a guidance vector field from a quantized voxel image, and deformation of an elastic surface within the vector field. The specific contributions are:

1. Generating approximate closed surfaces without holes from curve clouds with arbitrary topology
2. Preserving local smoothness while approximating the curve cloud
3. Enabling early and quick visualization of the curve cloud at early design stages for rapid evaluation and refinement
4. Bridging the gap in the design process by providing a means to automatically transform designers' raw, unrefined shape ideas into geometric data suitable for further beautification and refinement using conventional CAD tools

Currently, our approach creates a binary voxel image from the input 3D curve clouds. However, the nature of the sketching process typically results in curves that exhibit different levels of importance as often identified from the pressure intensity of the strokes. Our current drawing hardware is capable of recording such data, which can be used to generate gray scale voxel images. This will allow our vector field and resulting deformation algorithm to be more selective toward regions defined by heavily emphasized strokes.

Our studies have also indicated that in its current form, the user guided refinement operations described earlier may be difficult to master for naive users. The main difficulty arises in strategizing a sequence for which the successive operations result in the desired outcomes. Nonetheless, we have observed that users are able to adapt to the system relatively rapidly, after using the software on a handful of different examples.

As future work, we plan to improve both the computational efficiency of the guidance vector field calculation and the vector field resolution via adaptive discretizations such as octrees. We hope to achieve a better approximation capability at the vicinity of the curve clouds although field studies will be necessary to validate this need. As a continuation of this work, we are in the process of developing sketch-based surface modification tools which will allow designers to further refine the initial surfaces created by this work.

REFERENCES

[1] Igarashi, T., Matsuoka, S., and Tanaka, H., 1999. "Teddy: a sketching interface for 3d freeform design". In In Proceedings of SIGGRAPH 1999, ACM, p. 21.

[2] Nealen, A., Igarashi, T., Sorkine, O., and Alexa, M., 2007. "Fibermesh: designing freeform surfaces with 3d curves". *ACM Trans. Graph.,* ***26***(3), p. 41.

[3] Dekkers, E., Kobbelt, L., Pawlicki, R., and Smith, R. C., 2009. "A sketching interface for feature curve recovery of free-form surfaces". In 2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling, SPM '09, ACM, pp. 235–245.

[4] Sugihara, M., Groot, E. D., Wyvill, B., and Schmidt, R., 2008. A sketch-based method to control deformation in a skeletal implicit surface modeler.

[5] Schkolne, S., Pruett, M., and Schröder, P., 2001. "Surface drawing: creating organic 3d shapes with the hand and tangible tools". In Proceedings of the SIGCHI conference on Human factors in computing systems, CHI '01, ACM, pp. 261–268.

[6] Billinghurst, M., Grasset, R., and Looser, J., 2005. "Designing augmented reality interfaces". *SIGGRAPH Comput. Graph.,* ***39***, February, pp. 17–22.

[7] Kara, L. B., and Shimada, K., 2007. "Sketch-based 3d-shape creation for industrial styling design". *IEEE Comput. Graph. Appl.,* ***27***(1), pp. 60–71.

[8] Karpenko, O. A., and Hughes, J. F., 2006. "Smoothsketch: 3d free-form shapes from complex sketches". *ACM Trans. Graph.,* ***25***, July, pp. 589–598.

[9] Schmidt, R., Wyvill, B., Sousa, M. C., and Jorge, J. A., 2006. "Shapeshop: sketch-based solid modeling with blobtrees". In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, ACM.

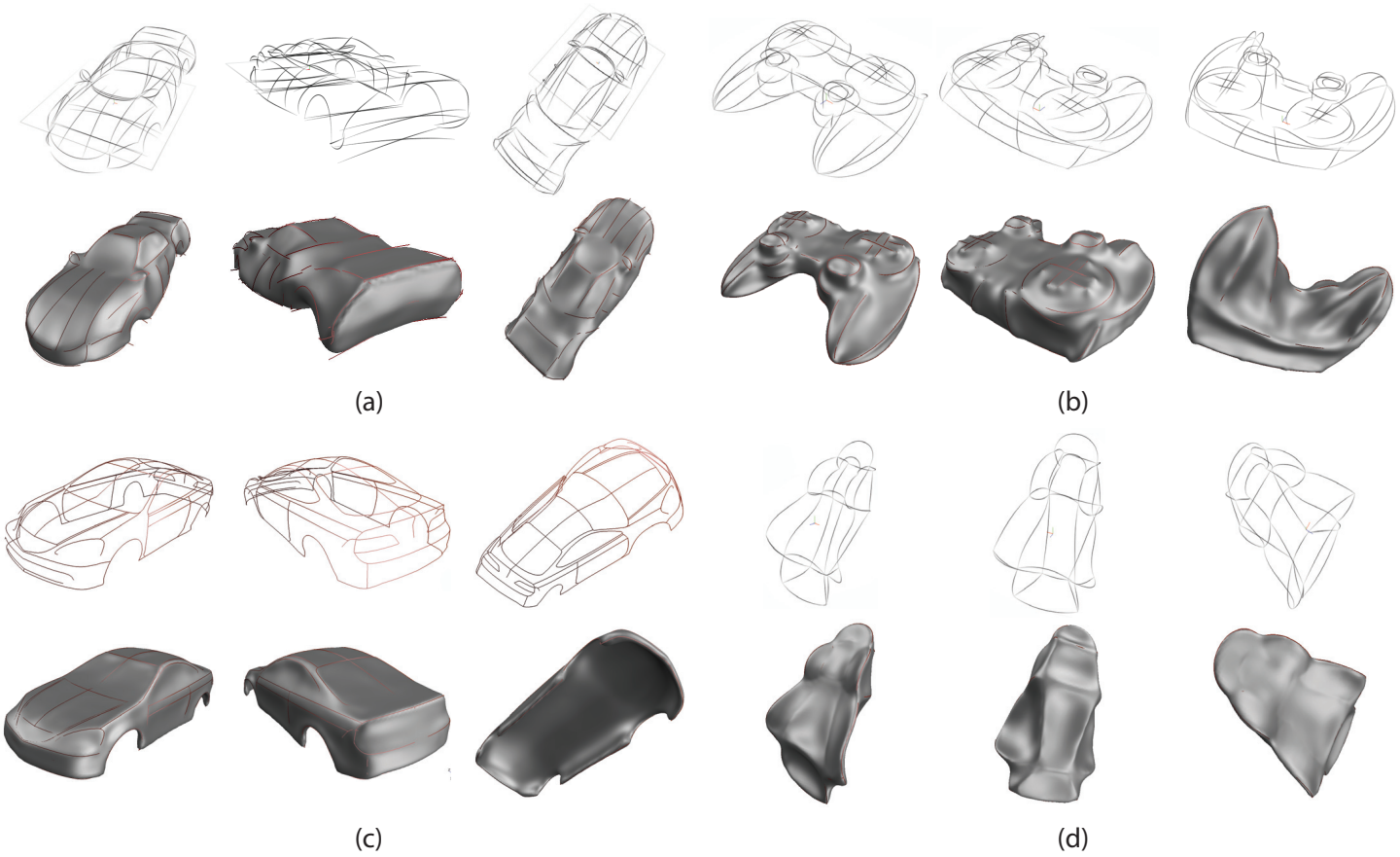


FIGURE 13. Different example cases for product form design, (a) concept car (b) gamepad (c) regular car (fully connected curve networks) (d) seat.

[10] Joshi, P., and Séquin, C., 2007. "C.: Energy minimizers for curvature-based surface functionals".

[11] Botsch, M., and Kobbelt, L., 2004. "An intuitive framework for real-time freeform modeling". *ACM Trans. Graph.,* ***23***, August, pp. 630–634.

[12] Gonzalez Castro, G., Ugail, H., Willis, P., and Palmer, I., 2008. "A survey of partial differential equations in geometric design".

[13] Zhang, J. J., and You, L. H., 2004. "Fast surface modelling using a 6th order pde". *Computer Graphics Forum,* ***23***(3), pp. 311–320.

[14] Barhak, J., and Fischer, A., 2001. "Parameterization for reconstruction of 3d freeform objects from laser-scanned data based on a pde method". *The Visual Computer,* ***17***, pp. 353–369. 10.1007/s003710100112354.

[15] Moreton, H. P., 1992. "Minimum curvature variation curves, networks, and surfaces for fair free-form shape design". PhD thesis, Berkeley, CA, USA. UMI Order No. GAX93-30652.

[16] Marinov, M., and Kobbelt, L., 2004. "Optimization techniques for approximation with subdivision surfaces". In Proceedings of the ninth ACM symposium on Solid modeling and applications, SM '04, Eurographics Association, pp. 113–122.

[17] van Overveld B. Wyvill, K., 15 JULY 2003. "Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface". *Visual Computer*.

[18] Lorensen, W. E., and Cline, H. E., 1987. "Marching cubes: A high resolution 3d surface construction algorithm". *SIGGRAPH Comput. Graph.,* ***21***, August, pp. 163–169.

[19] Xu, C., and Prince, J., 1998. "Snakes, shapes, and gradient vector flow". *Image Processing, IEEE Transactions on,* ***7***(3), Mar., pp. 359 –369.

[20] Bae, S.-H., Balakrishnan, R., and Singh, K., 2008. "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, UIST '08, ACM, pp. 151–160.

[21] Cohen, L., and Cohen, I., 1993. "Finite-element methods for active contour models and balloons for 2-d and 3-d images". *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* ***15***(11), Nov., pp. 1131 –1147.

[22] Cohen, L. D., 1991. "On active contour models and balloons". *CVGIP: Image Underst.,* ***53***, March, pp. 211–218.
[23] Meyer, M., Desbrun, M., Schröder, P., and Barr, A. H., 2002. "Discrete differential-geometry operators for triangulated 2-manifolds". *Visualization and mathematics,* ***3***(7), pp. 1–26.
[24] Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K., 1987. "Elastically deformable models". *SIGGRAPH Comput. Graph.,* ***21***, August, pp. 205–214.
[25] Yamada, A., Furuhata, T., Shimada, K., and Hou, K., 1999. "A discrete spring model for generating fair curves and surfaces". *Computer Graphics and Applications, Pacific Conference on,* ***0***, p. 270.
[26] Loop, C., 1987. *Smooth subdivision surfaces based on triangles*. Dept. of Mathematics, University of Utah.