A Technique for Constructing Developable Surfaces

Meng Sun
Eugene Fiume

Department of Computer Science
University of Toronto
10 King's College Road
Toronto, Canada, M5S 1A4

Abstract

Paper, sheet metal, and many other materials are approximately unstretchable. The surfaces obtained by bending these materials can be flattened onto a plane without stretching or tearing. More precisely, there exists a transformation that maps the surface onto the plane, after which the length of any curve drawn on the surface remains the same. Such surfaces, when sufficiently regular, are well known to mathematicians as *developable surfaces*. While developable surfaces have been widely used in engineering, design and manufacture, they have been less popular in computer graphics, despite the fact that their isometric properties make them ideal primitives for texture mapping, some kinds of surface modelling, and computer animation. Unfortunately, their constrained isometric behaviour cuts across common surface formulations. We formulate a new developable surface representation technique suitable for use in interactive computer graphics. The feasibility of our model is demonstrated by applying it to the modelling of a hanging scarf and ribbons and bows. Possible extensions and interesting areas of further research are discussed.

Keywords: computer-aided geometric design, surface modelling, developable surface, surface flattening

1 Introduction

The study and use of developable surfaces has a long history [4]. Real developable surfaces have natural applications in many areas of engineering and manufacturing. For instance, an aircraft designer uses them to design the airplane wings, and a tinsmith uses them to connect two tubes of different shapes with planar segments of metal sheets. In computer graphics, we are interested in modelling and animating objects seen in everyday life, and many objects can be approximated by piecewise continuous developable surfaces. Our aim is to work directly with developable surfaces as first-class modelling primitives for computer graphics. Our focus in this work is on the application of the theory of developable surfaces to the interactive creation of simple geometric models.

Developable surfaces may be deformable, but they have strong isometric properties. Because they can be easily parameterised so as to preserve arc lengths, they are excellent candidates for texture mapping [2][12]. However, developable surfaces are a small subclass of the polynomial or algebraic surfaces.

When manipulating surfaces defined using common piecewise polynomial surface formulations, it is easy to violate isometry properties. For instance, if we simulate the tearing of a piece of paper using an elastic model, the corresponding surface defined using a conventional formulation would appear to stretch and shear unnaturally during the tearing process. In general, it seems fruitful to devise modelling systems that are able to represent developable surfaces directly, and provide manipulation techniques that preserve their isometric properties.

In Section 2, we define and introduce properties of developable surfaces. In Section 3, we describe our new developable surface modelling technique. In Section 4, we present a hanging scarf and a bow modelled using our new approach. In Section 5, we discuss possible extensions to the system and areas of further research.

2 Developable Surface Modelling

Any surface whose (Gaussian) curvature vanishes at every point can be constructed by bending a planar region. These are *developable surfaces*. By their definition and their intrinsic properties, developable surfaces can be flattened onto a plane without stretching or tearing. In theory, the length of any curve drawn on such a surface remains the same, and the area of the developable surface also remains the same [6]. This is to say that curves on developable surfaces admit an easy arc-length parameterisation, and subregions of these surfaces have a direct surface-area parameterisation.

2.1 Definitions

We take our definitions of ruled and developable surfaces from [4]. Let $I = [a, b]$ be a closed real domain. A one-parameter family of lines $\{\boldsymbol{\alpha}(t), \boldsymbol{\beta}(t) : t \in I\}$ for a differentiable space curve $\boldsymbol{\alpha}(t)$ and a vector field $\boldsymbol{\beta}(t)$ is a correspondence that assigns to each $t \in I$ a point on $\boldsymbol{\alpha}(t) \in \mathbb{R}^3$ and a vector $\boldsymbol{\beta}(t) \in \mathbb{R}^3$, $\boldsymbol{\beta}(t) \neq \mathbf{0}$. For each $t \in I$, the line L_t passing through $\boldsymbol{\alpha}(t)$ that is parallel to $\boldsymbol{\beta}(t)$ is called *the line of the family at* t. For a one-parameter family of lines $\{\boldsymbol{\alpha}(t), \boldsymbol{\beta}(t)\}$, the surface

$$\mathbf{X}(t, v) = \boldsymbol{\alpha}(t) + v\boldsymbol{\beta}(t),\ t \in I,\ v \in \mathbb{R}, \tag{1}$$

is the *ruled surface* generated by that family. The lines L_t are the *rulings*, and the curve $\boldsymbol{\alpha}$ is a *directrix* of the surface $\mathbf{X}$. A ruled surface is *developable* if

$$\left\langle \boldsymbol{\beta} \times \frac{d\boldsymbol{\beta}}{dt}, \frac{d\boldsymbol{\alpha}}{dt} \right\rangle \ = \ 0, \tag{2}$$

This is to say that $\boldsymbol{\beta}$, $d\boldsymbol{\beta}/dt$ and $d\boldsymbol{\alpha}/dt$ are coplanar for all points on the surface. The notation $< \cdot, \cdot >$ signifies the inner product operator. The simplest examples of developable surfaces are cylinders and cones, and the simplest non-developable surface is a sphere. A *generalized cone* is a ruled surface generated by a family $\{\boldsymbol{\alpha}(t), \boldsymbol{\beta}(t)\}$, $t \in I$, where $\boldsymbol{\alpha}(I)$ is contained in a plane P and the rulings L_t all pass through a point $p \notin P$.

Every surface enveloped by a one-parameter family of planes is a developable surface. Each plane in this family is tangent to such a surface along a line that is obtained as the limiting position of the line in which two neighbouring planes intersect. Since the totality of these straight lines covers the entire surface, as shown in Figure 1, these straight lines are called the *generators* of the surface. In some cases, the generators envelop a space curve at which the developable surface has a sharp edge called *the edge of regression* or *cuspidal edge*, as shown in Figure 2.

2.2 Previous Work

Engineers, mathematicians and computer scientists are interested in developable surface modelling from different perspectives. The modelling problems can be formulated in different ways.

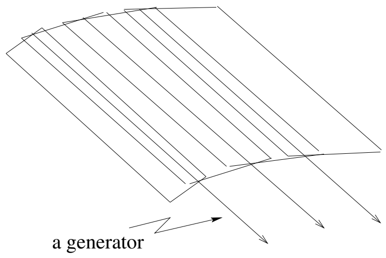


Figure 1: Generators of a developable surface.

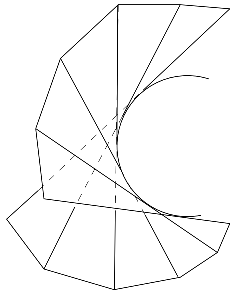


Figure 2: An edge of regression of a developable surface.

Given two distinct space curves, the classic problem of constructing a continuous developable surface connecting the two space curves has been extensively studied [5, 15]. A way is described to reconstruct a smooth developable surface from a given set of data points, where the data points are the spherical images of corresponding points of a geodesic of the original developable surface [11]. Redont approximates the spherical image of the geodesic, and builds a family of circular cones, each with a geodesic segment that corresponds to one segment of the original geodesic. Then Redont forms the desired developable surface using patches of the circular cones.

In [1], Aumann discusses a different developable surface modelling problem: given two distinct line segments x_1y_1 and x_2y_2 in $\mathbb{R}^3$ and a number of constraints, how do we determine a developable surface whose four boundaries are x_1y_1, x_2y_2, a Bézier curve with end points x_1 and x_2, and another Bézier curve with end points y_1 and y_2, provided that the constraints are satisfied. Aumann derives a number of theorems concerning the properties of developable surfaces under such constraints. Simulating the bending of a developable surface is also an interesting problem. In [8], Kergosien, Gotoga and Kunii studied the bending of a developable surface under external and internal forces.

In [3], Bodduluri and Ravani introduce a representation for developable surfaces in terms of plane geometry, using the concept of duality between points and planes in 3D projective space. The idea is to design a developable surface using control planes with appropriate basis functions. In [3], this approach is demonstrated using Bézier and B-spline bases. Geometric construction techniques, such as the de Casteljau and Farin-Boehm-type construction algorithms, are extended to the design of developable surfaces. The conversion from the dual form to a point representation is based on the line of regression.

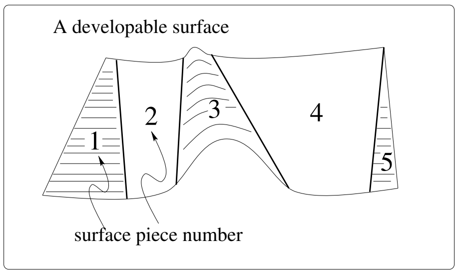
In [10], Pottmann and Farin present an approach to constructing Bézier and B-spline surfaces, based also on the dual representation in the sense of projective geometry. They transferred projective algorithms for NURBS curves to constructions for developable NURBS surfaces in dual rational B-spline form, using control planes, frame planes and two reference planes. The two reference planes are chosen dependent on the application. In [10], a new method for converting a dual NURBS surface to the usual NURBS tensor product form is discussed. This method takes advantage of fact that the planar intersection curve of a developable NURBS surface is a NURBS curve.

In this paper, we are primarily concerned with geometric considerations for modelling with developable surfaces, although we hope to generalise our work to physical models. Our new technique is inspired by the formulation of the classic problem and some ideas related to Redont's work.

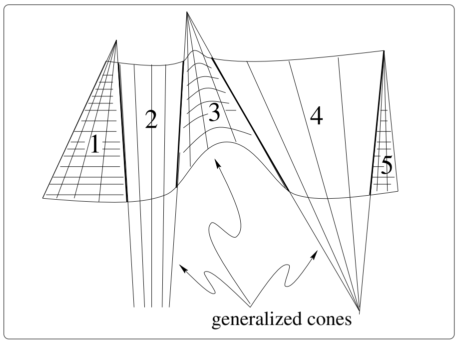
3 A New Modelling Primitive

Our new developable surface modelling technique reduces the geometric concept of a developable surface to a relatively simple visual specification. We use generalized cones to outline the shapes of segments of a developable surface. We divide the surface into several patches based on the geometry of the surface, as shown in Figure 3(a). In this paper, we restrict ourselves to G^1 continuous developable surfaces without cuspidal edges. Currently our modelling tool is set up to deal with sequential developable surface patches only, and it handles sheets with polygonal boundaries. We can easily extend the implementation to handle flattened sheets of arbitrary shape, but the underlying principle is clearer in the polygonal case. We approximate each patch by a generalized cone as shown in Figure 3(b). To define a generalized cone, we specify a cross section and the position of the apex in relation to this cross section as shown in Figure 3(c).

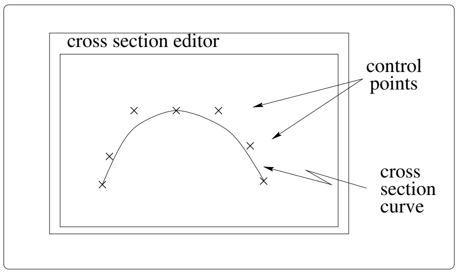
The specification of a piecewise G^1 developable surface is done at four levels of detail, as shown in Figure 4. From level 4 to level 1, the main task of constructing a developable surface is broken into more elementary subtasks. At level 4, a developable surface is desired. At level 3, the developable surface is divided into n surface patches. At level 2, each patch is associated with a generalized cone. The shape of a patch can be determined by the shape of the corresponding cone and the cut the curve makes through the cone. Several patches can be associated with the same generalized cone, possibly with different initial conditions. At level 1, each cone is specified by a cross section of the cone and the relative position of the apex and the cross section.



(a) Divide the surface into a number of pieces.



(b) Define the shape of each surface piece by a generalized cone.



(c) Use piecewise continuous Bézier curves to define a cross section of a generalized cone.

Figure 3: Dividing a developable surface into patches and defining each surface patch by a generalized cone.

3.1 Implementation

Using the data flow diagram shown in Figure 4, we can construct a developable surface in three steps.

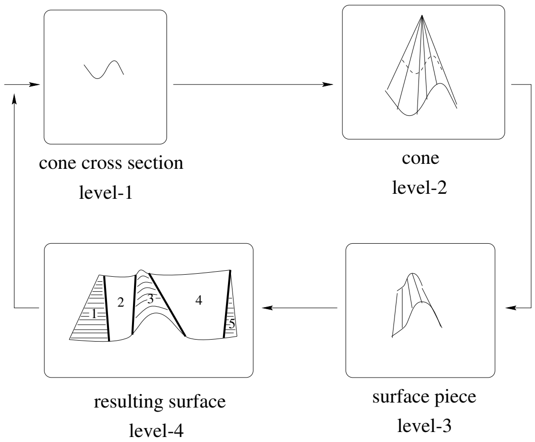


Figure 4: Flow of specification. The user specifies curve segments drawn from cross-sections of cones. The subsequent patches are inferred by the implementation.

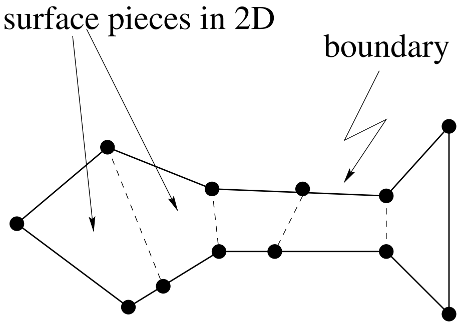


Figure 5: A 2D partitioning of a surface.

3.1.1 Surface Subdivision and Cone Constructions

First, the user observes the developable surface properties of the object to be modelled. Then the user has to decide on the subdivision of the surface. To specify the subdivision, the user needs to flatten the surface and inputs the coordinates of a sequence of points along the boundary of the surface in 2D. Then he/she can use these points to specify the partition of patches. An example is shown in Figure 5. When the surface is subdivided into n patches, we would use the shapes of generalized cones to outline the shapes of patches. For each patch, the user can define a cone by specifying a cross section and the position of the apex in relation to an interactively specified cross section as shown in Figure 3(c). The initial conditions will determine the region that needs to be trimmed from the generalized cone to obtain the patch, as we shall see in step 2. Up to this point, each cone is defined in its own 3D local coordinate system.

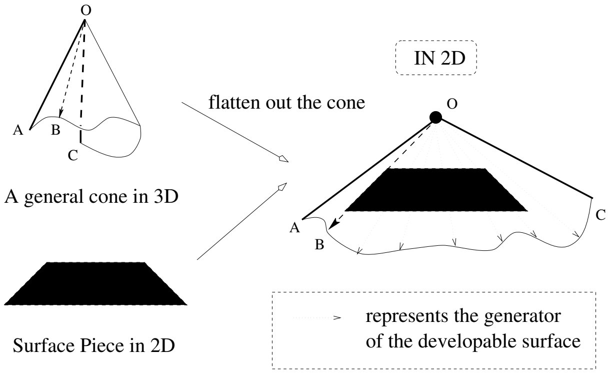


Figure 6: 2D map between a cone and a surface patch.

3.1.2 2D Mapping Between Cones and Surface Pieces

In this step, our task is to flatten out the user defined cones and determine the relationship between a patch and the corresponding flattened cone in 2D.

First, we want to flatten a user defined cone. As shown in Figure 6, when a cone is flattened out, for a given point on the surface patch, we can easily locate the generator passing through it. Clearly, when the cone is flattened out, the user-specified cross section corresponds to a plane curve. When dealing with a generalized cone, it is difficult to write down a closed form formula for this plane curve. However, we may use a system of differential equations and solve the problem numerically using a procedure described in [9]. Assume the curve is parametrized in terms of t. We can determine the profile of this plane curve using the system of differential equations

$$\begin{bmatrix} \dot{\psi}(t) \\ \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} \dot{s}(t)\kappa(t) \\ \dot{s}(t)\cos\psi(t) \\ \dot{s}(t)\sin\psi(t) \end{bmatrix}$$

along with the initial values $[\ \psi(0) \quad x(0) \quad y(0)\ ]^T = [\ \psi_0 \quad x_0 \quad y_0\ ]^T$, where $s(t)$ is the arc length of the plane curve, $\psi(t)$ is the angle subtended by the tangent of the plane curve on the x-axis as shown in Figure 7 [9]. Note that $\kappa(t)$ is the curvature at point $p(t)$ of the cross section of the cone in 3D, and the curvature at point $p(t)$ of a curve on the developed surface is equal to the curvature of the projection of the original curve onto the tangent plane of the surface at $p(t)$ [5].

Next, we determine the relationship between a patch and the corresponding flattened cone in 2D as shown in Figure 6. Without loss of generality, our system requires each patch to be defined as a convex polygon consisting of four vertices v_0, v_1, v_2 and v_3, where the line v_0v_3 is the first generator and the line v_1v_2 is the last generator of the

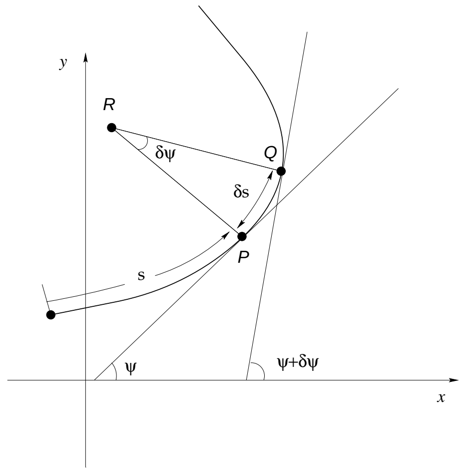


Figure 7: Define a 2D curve using s and ψ.

patch. Note that two of the four vertices might coincide. With the current version of our modelling system, the shape of each patch is outlined by one generalized cone. Surface patches are treated differently depending on their shapes, as given by their generators. Four general cases are considered:

CASE I:

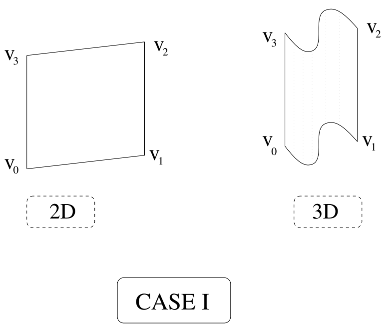


Figure 8: Surface patch type: Case I.

As shown in Figure 8, v_0v_3 is parallel to v_1v_2 in the plane embedding the flattened surface. In this case, the shape of the developable surface in 3D cannot be defined by a generalized cone, because after development we would still have $v_0v_3||v_1v_2$ in 3D. The shape of the developable surface in 3D can be defined by a generalized cylinder instead. In our system, this case is not currently implemented. To incorporate generalized cylinder into the model is easy, and it follows similar principles as those of generalized cones, i.e., flattening the cylinder out in 2D, finding the correspondence between points on a patch and points on the cylinder, and constructing the patch in 3D using this information.

CASE II: As shown in Figure 9, v_0v_3 and v_1v_2 inter-

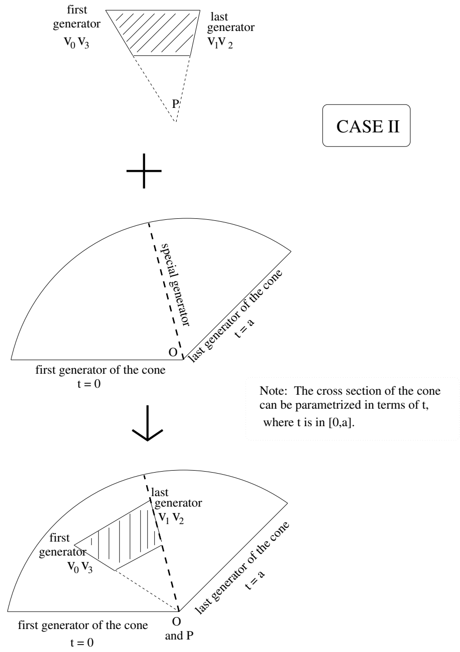


Figure 9: Surface patch type: Case II.

sect at a point p. In this case, the apex O of the corresponding cone has to coincide with the point p. This is because there is a one-to-one correspondence between generators of the patch and generators of the associated cone patch. Since the last generator v_0v_3 and the first generator v_1v_2 corresponds to two distinct generators of the cone, and two generators of a cone must intersect at the apex O of the cone, clearly, the apex O should coincide with the point p. The user can specify a special generator of the cone which corresponds to either v_0v_3 or v_1v_2 as one of the initial conditions mentioned in step 1. Then the relation between the cone and the patch can be determined as shown in the figure below.

CASE III: As shown in Figure 10, v_0, v_1 and v_2 are distinct vertices, and v_3 coincides with v_0. Since v_1v_2 is a generator, the apex of the cone has to be on the line defined by v_1v_2. Again, this is due to the correspondence between generators of the patch and generators of the corresponding cone. In this case, the user must specify a special generator of the cone corresponding to v_1v_2. When the user provides the distance from the apex to one of the vertices of the patch as one of the initial conditions mentioned in

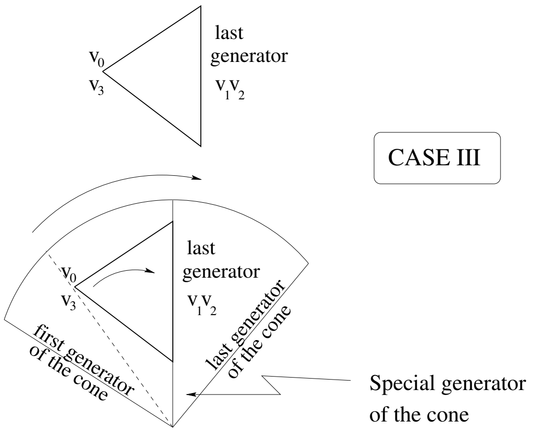


Figure 10: Surface patch type: Case III.

step 1, the relation between the patch and the cone can be determined as shown in Figure 10.

CASE IV: The case where v_0, v_1 and v_3 are distinct vertices, and v_2 coincides with v_1 is symmetric to case III.

3.1.3 Constructing a 3D Developable Surface

Using the 2D mapping shown in Figure 6, we can find, for each point on the surface patch, its counterpart on the cone. Since each cone is defined in its own 3D coordinate system, we can also represent the ribbon patch in this 3D local coordinate system.

Next, we position the surface patch in the 3D world coordinate system. Since currently the system is set up to deal with sequential developable surface patches, let the developable surface patch be numbered $0, 1, 2, ..., n-1$, and let the last surface generator of the i^{th} surface patch be the first surface generator of the $(i+1)^{st}$ surface patch.

To properly connect the two adjacent patches, we want the shared surface generator to be correctly aligned and the surface normals of each patch at that boundary to be parallel. A unique linear transformation matrix can be determined using these constraints.

Assume that surface patch 0 is already in its proper position. Our algorithm connects the i^{th} patch to the $(i-1)^{st}$ surface in the i^{th} iteration for $i = 1, 2, ..., n-1$, i.e., by the end of the i^{th} iteration, patches 0 through i are in their proper positions in the 3D world coordinate system. At the beginning of the i^{th} iteration, we consider patches i and $i-1$. We need some notation. In the world coordinate system, let AB be the final generator of patch $i-1$, the unit direction vector of AB is $\mathbf{g} = (g_x, g_y, g_z)$, the unit surface normal of patch $i-1$ along AB is $\mathbf{n}$, and the position of A is (a_x, a_y, a_z).

Similarly, in the local coordinate system of patch i, we label the first generator of patch i as $A'B'$, the unit direction vector of $A'B'$ is $\mathbf{g'} = (g'_x, g'_y, g'_z)$, the unit surface normal of patch i along $A'B'$ is $\mathbf{n'}$, and the position of A' is (a'_x, a'_y, a'_z).

To make the common generator AB align with $A'B'$ and make the unit surface normal along this shared generator equal, we can use the matrix M to transform patch i from its local coordinate system to its proper position in the world coordinate system, where

$$M = M_1 M_2 M_3 M_4,$$

$$\begin{aligned}
M_1 &= \begin{bmatrix} 1 & 0 & 0 & a_x \\ 0 & 1 & 0 & a_y \\ 0 & 0 & 1 & a_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
M_2 &= \begin{bmatrix} g_x & n_x & g_x \times n_x & 0 \\ g_y & n_y & g_y \times n_y & 0 \\ g_z & n_z & g_z \times n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
M_3 &= \begin{bmatrix} g'_x & n'_x & g'_x \times n'_x & 0 \\ g'_y & n'_y & g'_y \times n'_y & 0 \\ g'_z & n'_z & g'_z \times n'_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
M_4 &= \begin{bmatrix} 1 & 0 & 0 & -a'_x \\ 0 & 1 & 0 & -a'_y \\ 0 & 0 & 1 & -a'_z \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\end{aligned}$$

The output of this step is the resulting developable surface as a whole in 3D. This step is requires no user intervention. The resulting developable surface is G^1 continuous.

3.2 Shape Control

To modify or refine the shape of a patch, it is not generally necessary to subdivide it. The user can adjust and add control points of the cross section curve of the cone associated to the patch (cf. Figure 3). Since users can interactively change the piecewise continuous cross section curve of the generalized cone defining the shape of associated patch, users have control over the local shape of each patch. However, developable surfaces are not stretchable, and thus the adjustment of the shape of a patch might affect the relative orientation of this patch and its adjacent patch in object space.

Further control over the patches can be achieved by adjusting control points of cross section curves. For instance, suppose we have two adjacent patches A and B as shown in Figure 11(a). To control the relative orientation of patches A and B, we can change the orientation of

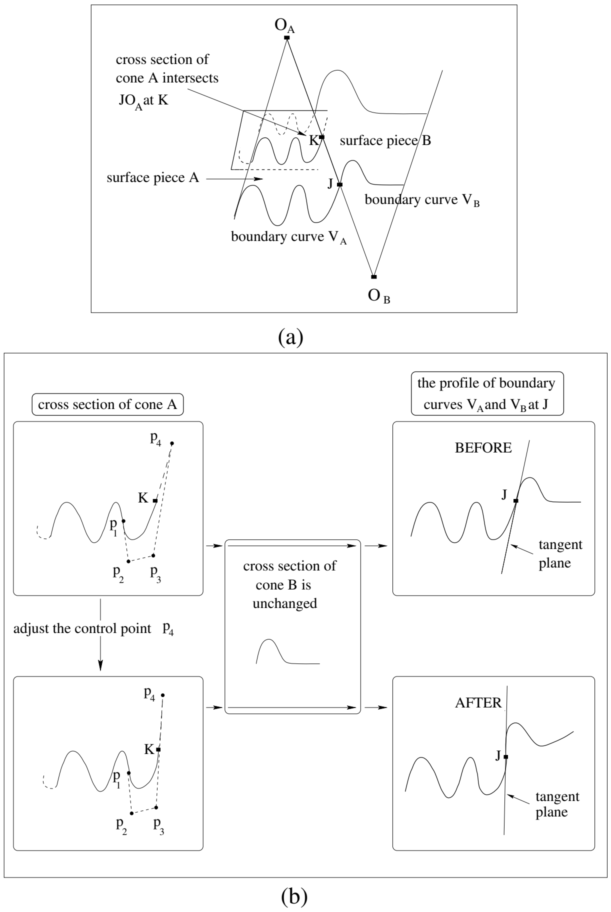


Figure 11: Adjusting one control point of a cross section curve to change the relative orientation of two patches A and B.

the tangent plane at their common generator. That can be done by adjusting a few Bézier control points of the user-defined cross section curve of cone A, where cone A is associated to patch A.

An example of a local adjustment and its overall effect is shown in Figure 11(b). The boundary curve V_A of A and the boundary curve V_B of B join each other at point J. The cross section of cone A is shown in the left column of Figure 11(b). Point K is the intersection point of generator JO_A and the cross section of cone A. Recall that the cross section curve is defined by piecewise cubic Bézier curves in our modelling system. Assume that point K is on the cubic Bézier defined by control points p_1, p_2, p_3 and p_4. When p_4 is adjusted, the tangent direction of the cross section curve at K might be changed as shown in the figure. As a consequence, the orientation of the tangent plane along the common generator O_AO_B is changed, as shown in the right column of Figure 11(b). Therefore, the relative orientation of patches A and B are changed, as one can tell from the profile of the boundary curves V_A and V_B at the neighbourhood of point J.

Note that the global shape of all surfaces is not largely affected by this adjustment. Since the cross section curve of cone A is defined using piecewise continuous Bézier curve segments, moving control point p_4 affects the shape of only the last segment of the cross section curve. A user can choose to make this segment relatively small so that it can be used for more global control, and he/she can still have local control over the shape of the resulting developable surface.

4 Applications of the Model

To demonstrate the feasibility of our approach, we will use our developable surface representation technique to model a hanging scarf and a looping ribbon structure.

4.1 Modelling a Hanging Scarf

Imagine lifting a silky scarf by a single corner, leaving the scarf hanging naturally in the air. Such a hanging scarf is approximately a developable surface. Given a point on the scarf's surface, one can easily approximate where the generator lies. We can imagine an "invisible" generalized cone of a similar shape suspended in the air by its apex and the cross section of the cone as is shown in Figure 12(a). The corner by which the scarf is suspended is close to the apex of the cone, and the scarf itself follows the contour of the cone.

The folds produced by the cross section, as shown in Figure 12(a), are expected to be very close to one another. In order to give the reader a better idea what the folds look like we will use the cross section in Figure 12(b) to demonstrate the design and modelling process in the following sections.

4.1.1 Approximating a Single Cross Section

By inspection, we can obtain some intuition for the folding of the hanging scarf. The point of departure for our system is an estimate what the horizontal cross section of the scarf looks like. The user designs the cross section in an interactive Cross Section Editor using cubic Bézier curves. The user may add or move Bézier control points. A neighbouring point can be adjusted automatically by the system, to make sure that the resulting piecewise curve is C^1 continuous.

The approximated horizontal cross section of the scarf is used as the cross section of the "invisible" cone. The apex position of the cone in 3D is specified in a script file.

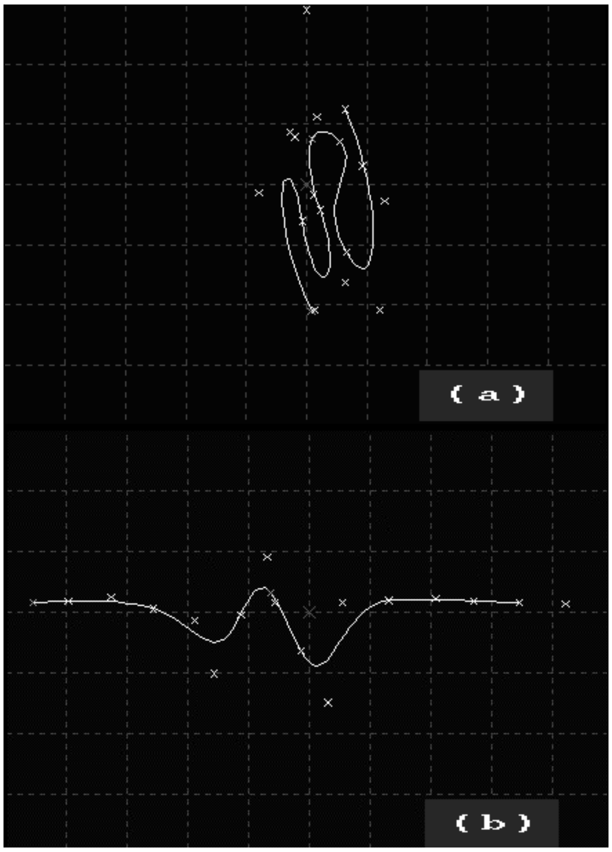


Figure 12: Two possible cross section designs (Hanging Scarf).

4.1.2 Defining Surface Patches in 2D

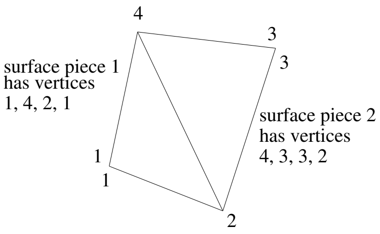


Figure 13: Surface patch partition (Hanging Scarf).

Given the four vertices, the scarf is divided into two patches, as shown in Figure 13. The situations described in case III and case IV in section 3.1.2 occur in our hanging scarf example. The same generalized cone is used for both patches.

4.1.3 Constructing Two Surface Pieces in 3D

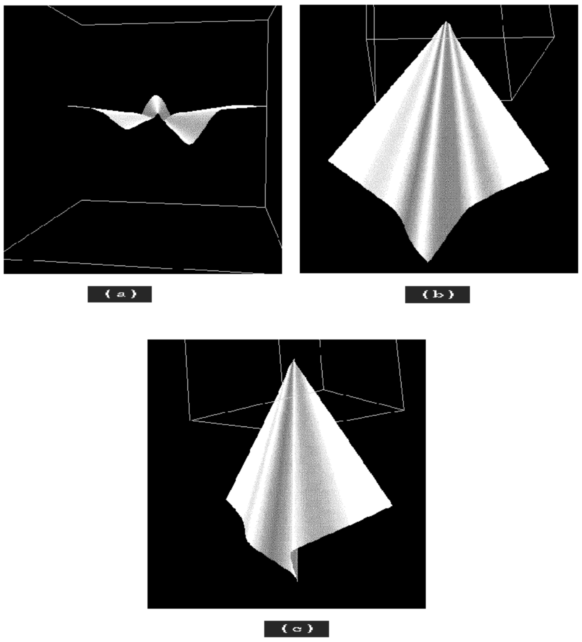


Figure 14: A hanging scarf: (a) top view; (b) front view; (c) view from an angle.

We have discussed the method of constructing the developable surface in 3D in section 3. The resulting surface is as shown in Figure 14. The configurations are: (a) the top view of the scarf, (b) the front view of it, (c) the view from an angle.

4.2 Modelling a Looping Structure

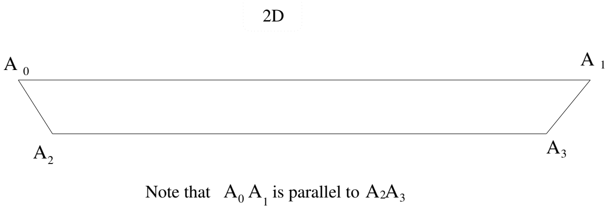


Figure 15: A long ribbon.

The looping structure example used in this section is a bow which resembles the type of bow used to decorate a gift box. We will "fold" a long ribbon into a bow, and the ribbon used here is as shown in Figure 15.

There is a pattern in the structure of a bow. If we use an ellipse of roughly the same size to replace a loop in a bow, the pattern becomes more obvious. Some example patterns are shown in Figure 16. It seems sensible to define the shape of each loop using a generalized cone,

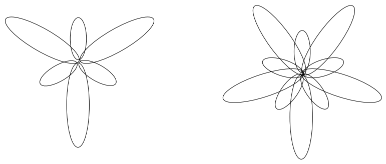


Figure 16: Example bow patterns.

and to introduce an intermediate patch to connect two adjacent loops if necessary.

4.2.1 Approximating the Cross Section

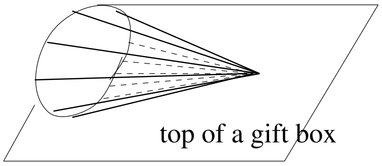


Figure 17: A cone used to define a loop.

To define a loop, one might use a cone which looks like the one shown in Figure 17. Here, we make the portion which is flattened as it touches the top of a gift box flatly, so that the resulting loop looks more realistic. We have

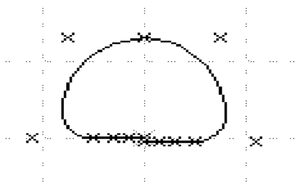


Figure 18: The **cross section** of the cone used for a loop.

constructed a cone cross section accordingly as shown in Figure 18. The intermediate patches are triangular in 3D in this example, so the cone that defines their shape uses a straight line as its cross section.

4.2.2 Constructing the 2D Layout

In this example, we will create three small loops and three big loops in our bow. We take a long ribbon, divide it into ten segments – six loop segments and four intermediate connecting segments. We draw the ribbon in the 2D layout window, rotate the 2D mapping of a cone to match its corresponding ribbon patch. Figure 19(b) is an overall picture of the ribbon patches and the associated cones in 2D. Figure 19(a) shows some details of the 2D layout. The step by step construction of the 2D layout is as shown in Figure 20.

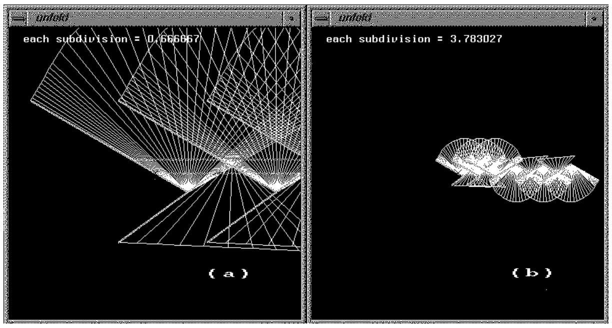


Figure 19: The 2D layout of the bow.

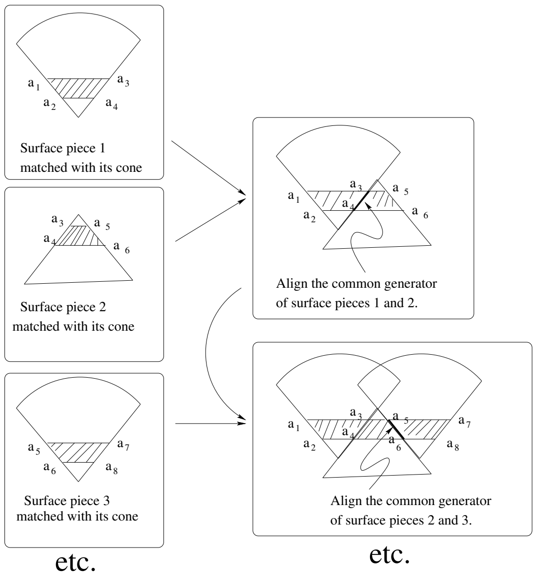


Figure 20: Step by step construction of the 2D layout of the looping structure.

4.2.3 Constructing the 3D Layout

We have discussed the method of constructing the developable surface in 3D in the section 3 . Figure 21 shows the development of a surface in 3D.

5 Conclusions

Developable surfaces are a small subclass of the parametric surfaces, but it is important to explore how far they can be taken as full-fledged modelling primitives. In this paper, we have considered a relatively simple case of creating sequences of piecewise continuous developable surface patches. Immediate extensions to more general grids of patches are required. Developable surfaces defined over triangular domains also warrant exploration, although this may require some reformulation of our work.

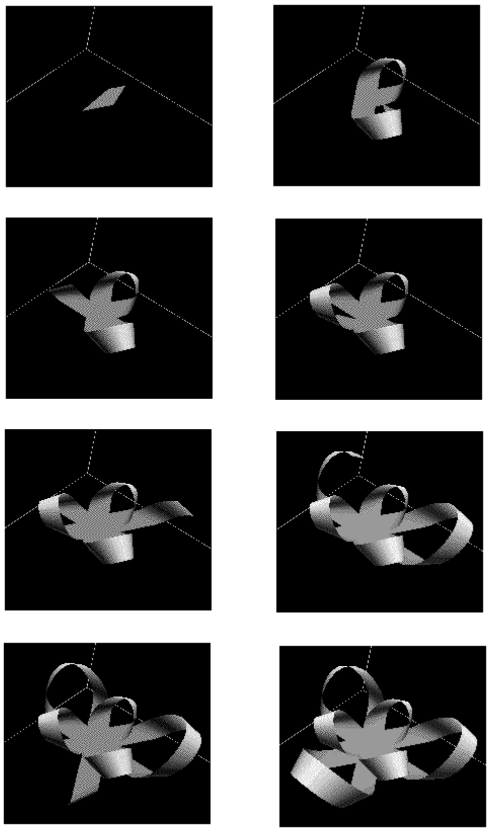


Figure 21: The development of the bow in 3D.

It is becoming common to superimpose a physical interpretation over a geometric formulation. It would be of great interest, for example, to allow our hanging scarf to respond accurately to physical forces while still maintaining a developability constraint.

6 Acknowledgements

We gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada, the Ontario Information Technology Research Centre, and the University of Toronto.

References

[1] G. Aumann, "Interpolation with Developable Bézier Patches", *Computer Aided Geometric Design*, Vol. 8, (1991), pp 409-420.

[2] C. Bennis, J.M. Vezien, and G. Iglesias, "Piecewise Flattening for Non-distorted Texture Mapping", *Proceedings of ACM SIGGRAPH '91*, (July, 1991), pp 237-246.

[3] R.M.C. Bodduluri and B. Ravani, "Design of Developable Surfaces Using Duality Between Plane and Point Geometrie s", *Computer-Aided Design 25*, (October, 1993), pp 621-632.

[4] M. Do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice Hall, Englewood Cliffs, NJ, U.S.A., 1976.

[5] I. D. Faux and M. J. Pratt, *Computational Geometry for Design and Manufacture*, Ellis Horwood, Chichester, UK, 1980.

[6] D. Hilbert and S. Cohn-Vossen, *Geometry and the Imagination*, Chelsea, New York, NY, U.S.A., 1952.

[7] J. Hoschek and D. Lasser, translated by Larry L. Schumaker, *Computer Aided Geometric Design*, A K Peters, Wellesley, Massachusetts, 1993.

[8] Y. L. Kergosien, H. Gotoda and T. L. Kunii, "Bending and Creasing Virtual Paper", *IEEE Computer Graphics and Applications* (Jan. 1994), pp 40-48.

[9] A. W. Nutbourne, P. M. McLellan and R. M. L. Kensit, "Curvature Profiles for Plane Curves", *Computer-Aided Design*, Vol. 4, No. 4, (July, 1992), pp 176-184.

[10] Helmut Pottmann and Gerald Farin, "Developable Rational Bézier and B-spline Surfaces", *Computer Aided Geometric Design 12*, (August, 1995), pp 513-531.

[11] P. Redont, "Representation and Deformation of Developable Surfaces", *Computer Aided Design*, Vol. 21, No. 1, (Jan/Feb, 1989), pp 13-20.

[12] J. Maillot, J. Yahia, and A. Verroust, "Interactive Texture Mapping", *Proceedings of ACM SIGGRAPH '93*, (August, 1993), pp 27-34.

[13] P. Volino, M. Courchesne and N. Magnenat Thalmann, "Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects", *Proceedings of ACM SIGGRAPH '95*, (August, 1995), pp 137-144.

[14] A. Watt and M. Watt, *Advanced Animation and Rendering Techniques*, Addison-Wesley, New York, N.Y., 1992.

[15] G. Weiss and P. Furtner, "Computer-aided Treatment of Developable Surfaces", *Computer & Graphics*, Vol. 12, No. 1, (1988), pp 39-51.