Regular Paper

Surface Deformation by Surface Transformations

NAOKI URANO*

This paper presents a surface transformation technique that can be applied to most currently used surface models, such as polygon models and parametric surfaces. A surface is deformed by applying surface transformation, which is defined by a pair of surfaces. A surface with a complicated shape can be created by combining simple surface deformations. In this technique, deformation is independent of the geometric model. The operation is very graphical and provides designers with an intuitive interface. Several examples are presented.

1. Introduction

Although rendering techniques for making pictures in computer graphics have improved significantly in the last ten years, a tremendous amount of time is still required in order to input general geometric data, especially free-form objects. The editing of geometric objects, called deformation, is even more difficult. User-computer interactions should be easy for humans, rather than for computers, to understand. This means that the interface should be intuitive for humans [5], a requirement that has not yet been satisfied in 3D geometric modeling and the deformation of 3D objects. The ultimate solution will be physically based modeling [16, 19, 20], an approach in which the model itself behaves like the real objects modeled, but the computational cost of this approach is currently too high for it to be used in interactive environments. In this paper, we discuss only surface deformation, because surface models, such as polygon models, B-spline surfaces, Bézier surfaces, and NURBS, have been widely used in industry for a fairly long time. In addition, deformation is an important key to input in the interactive modeling of 3D objects. A designer usually refines an object progressively in the design process. This process, which includes deformation, is a very important part of a designer's work [9].

Surface deformation has been studied for a long time. There are two major issues here: intuitive interface and model independence. As mentioned at the beginning of the section, interface is very important. Any 3D object shapes can be created if the appropriate geometric data are carefully specified one by one. But this is very cumbersome. An intuitive interface implies various features, of which one of the most important is that the interface should be graphical. Parent. et al. used the sculpturing metaphor for surface deformation [15, 14]. This is easy for designers to understand, since sculpturing a 3D object with a tool is a natural interface. Bartels' sweeping method can be used to design a free-form surface by associating curves [3]. Bloomenthal, et al. proposed convolution surfaces [4]. These approaches of combining graphic objects are easy for designers to understand, since a surface is specified by curves and surfaces.

The second important issue is model independence. A deformation should be defined independently of the surface model. Designers often complain that the operation of deformation is too mathematical. In other words, it is very dependent on the geometric model. For example, a parametric surface consists of a network of control points, and deformation is performed by moving these control points. Although this approach is fairly graphical and reasonably understandable, the range of deformation is very dependent on the surface model. Barr proposed an elegant deformation method for solid objects that is also applicable to surface models [1]. Sclaroff extended the idea and proposed a general formula for graphics models [17]. Sederberg's Free-Form Deformation (FFD) and its extension, Extended Free-Form Deformation (EFFD), proposed by Coquillart, are very powerful tools and can be applied to any geometric models [18, 7, 8]. However, it is sometimes difficult to find the values of the geometric coordinates in the parametric space when FFD or EFFD is applied to already existing 3D objects. Another important requirement in terms of model independence is to provide various granularities for deformation control. It is necessary to allow a design process that moves from a rough design to fine details. Forsey, et al. solved the problem by using hierarchical surfaces [6, 13].

This paper proposes an alternative technique for surface deformation. The purpose of the technique is to satisfy some of the requirements discussed above, namely that

*IBM Research, Tokyo Research Laboratory, IBM Japan Ltd. 5-19, Sanbancho, Chiyoda-ku, Tokyo 102, Japan.

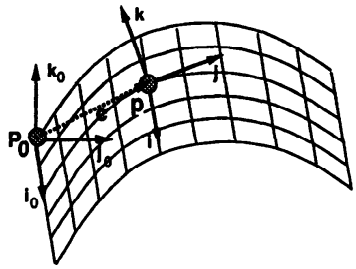

Fig. 1 Surface Point.

- Deformation should be specified graphically.
- Deformation should be specified independently of the surface to be deformed.
- Deformation should be performed hierarchically.

2. Surface Matrices

This section describes surface matrices. A surface matrix is defined by a surface. This definition is necessary since a surface transformation, which is described in the next section, is derived from a pair of surfaces. Most currently used tensor product parametric surfaces can be represented in this manner.

We assume that a surface exists in three-dimensional space, $\mathscr{R}^3$, and is represented as a group of surface points in $\mathscr{R}^3$. Each point on the surface has its own coordinate system in $\mathscr{R}^3$. We call such a coordinate system a frame. Let (i, j, k) be a representation of the orientation of a frame. Each component is a 3D vector in this case. For example, the vector i has the components x, y, and z. Figure 1 shows the reference frame (i_0, j_0, k_0), a point P on the surface, and its orientation (i, j, k). The position of P is expressed by a 3D vector from the origin of the reference frame (i_0, j_0, k_0), P_0. Let this vector be c. Figure 1 shows the vector c of the point P, which is expressed by the vector from P_0 to P. Given the vector c, the frame of the point is represented as (i, j, k, c). Thus, the surface is defined as a group of points (i, j, k, c). Assuming that the surface has some parameters, and letting α be a representation of the parameters, we can represent the surface matrix $S(\alpha)$ by the following equation:

$$S(\alpha)=\begin{pmatrix} i(\alpha) & 0 \\ j(\alpha) & 0 \\ k(\alpha) & 0 \\ c(\alpha) & 1 \end{pmatrix} \tag{1}$$

In this case, the matrix is 4×4.

3. Surface Transformation

This section describes transformation from one surface to another. As described in the previous section, a point on a surface can be represented in matrix form. A point (i_1, j_1, k_1, c_1) on one surface is transformed into a point (i_2, j_2, k_2, c_2) on another surface by a transformation matrix M. This is represented by the following equation:

$$\begin{pmatrix} i_1 & 0 \\ j_1 & 0 \\ k_1 & 0 \\ c_1 & 1 \end{pmatrix} M=\begin{pmatrix} i_2 & 0 \\ j_2 & 0 \\ k_2 & 0 \\ c_2 & 1 \end{pmatrix} \tag{2}$$

Given two surface matrices, $S_1(\alpha)$ and $S_2(\beta)$, where α and β are the parameters of surfaces S_1 and S_2, respectively, if there is one-to-one mapping between α and β, then $S_1(\alpha)$ is transformed into $S_2(\beta)$ by a matrix, $M(\alpha, \beta)$. We call the matrix $M(\alpha, \beta)$ the surface transformation matrix. The mapping from a point of S_1 to a point of S_2 is one-to-one, and is uniquely determined. The surface transformation is represented by the following equation:

$$S_1(\alpha)M(\alpha, \beta)=S_2(\beta) \tag{3}$$

where

$$S_1(\alpha)=\begin{pmatrix} i_1(\alpha) & 0 \\ j_1(\alpha) & 0 \\ k_1(\alpha) & 0 \\ c_1(\alpha) & 1 \end{pmatrix} \tag{4}$$

and

$$S_2(\beta)=\begin{pmatrix} i_2(\beta) & 0 \\ j_2(\beta) & 0 \\ k_2(\beta) & 0 \\ c_2(\beta) & 1 \end{pmatrix} \tag{5}$$

Figure 2 shows that point P_1 of surface S_1 is transformed into point P_2 of surface S_2 by transformation M.

The transformation matrix is computed by multiplying both sides of Equation 3 by the inverse of the surface matrix, S_1^{-1}, as follows:

$$M(\alpha, \beta)=S_1^{-1}(\alpha)S_2(\beta) \tag{6}$$

If surface S_1 has the parameters α, then the inverse matrix S_1^{-1} in Equation 6 has the parameters α. Hence, the transformation matrix $M(\alpha, \beta)$ can be computed when the values of α and β are determined.

4. Surface Deformation

This section explains a surface deformation technique. Deformation is performed by applying surface transformation to a surface. Assume that the surface to be deformed, S_3, has parameters γ. Given the surface matrix, $S_3(\gamma)$, and the surface transformation matrix,

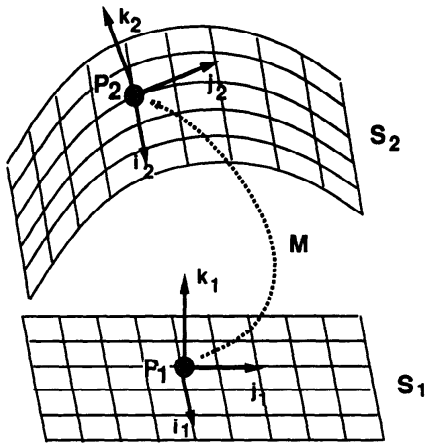

Fig. 2 Surface Point Mapping.

$M(\alpha, \beta)$, and assuming that α, β, and γ have one-to-one mapping with respect to one another, we can determine the deformed surface, $S_4(\alpha, \beta, \gamma)$, as follows:

$$S_4(\alpha, \beta, \gamma) = S_3(\gamma) M(\alpha, \beta) \tag{7}$$

Figure 3 shows that surface S_3 is deformed to S_4 by the surface transformation M, which is derived from the given surfaces, S_1 and S_2. Applying the same surface transformation M to the deformed surface S_4 yields surface S_5, which is more bent.

However, the reference frame of surfaces S_1 and S_2 may not be always the same as the reference frame of surface S_3. Hence, the following transformation matrix, R, should also be introduced:

$$S_4(\alpha, \beta, \gamma) = S_3(\gamma) R M(\alpha, \beta) \tag{8}$$

Furthermore, the deformation frame to which the deformation transformation applies may be changed from one point to another on the surface. In this case, the transformation matrix R has various values on the surface, and Equation 8 becomes as follows:

$$S_4(\alpha, \beta, \gamma) = S_3(\gamma) R(\gamma) M(\alpha, \beta) \tag{9}$$

5. Examples

This section examines the case of tensor product parametric surfaces, such as B-spline surfaces and Bézier surfaces. A tensor product parametric surface is defined by $m \times n$ control points, b, and by piecewise polynomial basis functions, $B_i^m(u)$ and $B_j^n(v)$, of degrees m and n, respectively [2, 11, 12]. In the case of Bézier surfaces, the basic functions are Berstein polynomials. The parameters u and v vary independently. A point on the the surface, c, is

$$c(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{i,j} B_i^m(u) B_j^n(v) \tag{10}$$

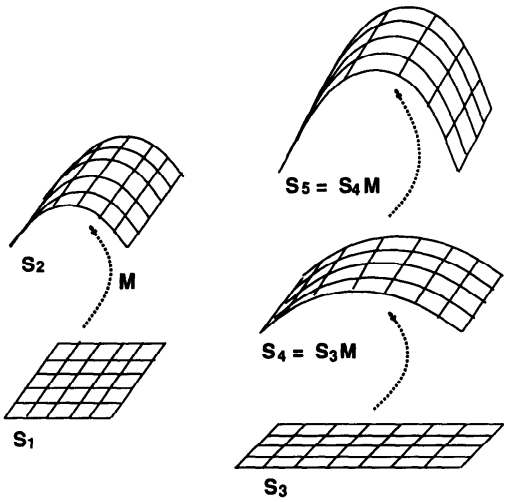

Fig. 3 Surface Deformation.

The unit tangent vectors on the surface are computed by taking partial derivatives of the surface equation as follows [10]:

$$t_u(u, v) = \frac{\frac{\partial}{\partial_u} c(u, v)}{\left| \frac{\partial}{\partial_u} c(u, v) \right|} \tag{11}$$

and

$$t_v(u, v) = \frac{\frac{\partial}{\partial_v} c(u, v)}{\left| \frac{\partial}{\partial_v} c(u, v) \right|} \tag{12}$$

The normal vector of the surface is the cross-product of the tangent vectors at the point c [10]. The equation of the normal vector is

$$n(u, v) = \frac{\frac{\partial}{\partial_u} c(u, v) \times \frac{\partial}{\partial_v} c(u, v)}{\left| \frac{\partial}{\partial_u} c(u, v) \times \frac{\partial}{\partial_v} c(u, v) \right|} \tag{13}$$

Assuming that the two tangent vectors are orthogonal, the frame of the surface can be represented by the tangent vectors and the normal vector. Thus the surface is represented as follows:

$$S(u, v) = \begin{pmatrix} t_u(u, v) & 0 \\ t_v(u, v) & 0 \\ n(u, v) & 0 \\ c(u, v) & 1 \end{pmatrix} \tag{14}$$

Let us take a very simple case of deformation, the one that appears in Fig. 3. Surface S_1 is flat and square and surface S_2 is generated by bending surface S_1. Since

surface S_1 is a square mesh, it is represented as follows:

$$S_1(u_1, v_1)=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ & c_1(u_1, v_1) & & 1\end{pmatrix} \quad (15)$$

and its inverse is:

$$S_1^{-1}(u_1, v_1)=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ & -c_1(u_1, v_1) & & 1\end{pmatrix} \quad (16)$$

S_2 is:

$$S_2(u_2, v_2)=\begin{pmatrix}t_{2_{u_2}}(u_2, v_2) & 0\\ t_{2_{v_2}}(u_2, v_2) & 0\\ n_2(u_2, v_2) & 0\\ c_2(u_2, v_2) & 1\end{pmatrix} \quad (17)$$

Thus the transformation matrix M in Equation 6 is:

$$M(u_1, v_1, u_2, v_2)=S_1^{-1}(u_1, v_1)S_2(u_2, v_2) \quad (18)$$

If the reference frame of the surface S_3 that is to be deformed is the same as the reference frame of surface S_1, then the transformation matrix R in Equation 9 is the identity matrix. Thus, surfaces S_4 and S_5 are represented as the following matrices:

$$S_4(u_1, v_1, u_2, v_2, u_3, v_3)=S_3(u_3, v_3)M(u_1, v_1, u_2, v_2) \quad (19)$$

$$S_5(u_1, v_1, u_2, v_2, u_3, v_3) =S_4(u_1, v_1, u_2, v_2, u_3, v_3)M(u_1, v_1, u_2, v_2) \quad (20)$$

Figure 4 shows surface S_1 and Fig. 5 shows the deformation surface S_2 in relation to surface S_1. Given these surfaces, the transformation matrix can be computed as previously described. The surface S_3 in Fig. 6 is deformed to the surface S_4 in Fig. 7 by the transformation matrix. Applying the same deformation to the surface of Fig. 7 yields the surface S_5 shown in Fig. 8. Figure 10 shows the result when the same deformation is applied to the bumpy surface in Fig. 9. Figure 12 shows the result when the same deformation is applied to the wavy surface of Fig. 11. Another example of a deformation surface is shown in Fig. 13. The deformation is defined in relation to surface S_1. Figure 14 shows the result when the deformation is applied to the surface of Fig. 11. Figure 15 shows an example of combination of deformations.

In the example, the inverse matrix can be evaluated when the parameters u_1 and v_1 are determined. However, in more general cases, it is sometimes necessary to compute the inverse matrix at each sample point. This often happens when S_1 has a more complicated shape than a simple plane. It is not usually intuitive to imagine a deformation from a complicated shape to a simple shape, and this is not our intention with regard to deformation.

Note that all the surfaces appearing in the figures are evaluated at arbitrary sample points on the surfaces and are tessellated to triangles when they are rendered in Phong shading. Polygon tessellation is a practical technique when parametric surfaces are rendered in an interactive system, because it is computationally expensive to render them directly, especially in the case of parametric surfaces with many parameters.

6. Conclusion

This paper has presented a surface deformation technique. We showed that a deformation can be specified graphically by a pair of surfaces, independently of the surface to be deformed. Thus, many types of deformation could be predefined as deformation tools. It is not necessary for users to know the mathematical elements behind the deformation tool. They only need to recognize the deformation from the shapes of a pair of surfaces. Users can deform surfaces with these visible deformation tools. This graphical specification increases the intuitiveness of the interface. The definition of surfaces also reduces the number of data required, because a deformation itself is usually defined by simple surfaces.

The surface model used in the definition of deformation can be different from the surface model of the object to be deformed. For example, a deformation can be defined by a pair of Bézier surfaces when the surfaces to be deformed are B-spline surfaces. Therefore, it is independent of the surfaces to be deformed. Although only the tensor product parametric surface case was discussed in this paper, the basic idea can be applied to other surface models as long as the geometric characteristics (the normals and orthogonal tangents) and the one-to-one mappings among the surfaces are known. However, in some surface models, it is hard to find them. For example, in the case of a polygon model, it would be relatively easy to find the geometric characteristics, because the model often includes enough information, but it might be difficult to find the one-to-one mapping.

We also showed that a surface with a complicated shape can be created by combining deformations. The deformed surface is not converted into a normal surface model. The final deformed surface is expressed in the original surface and a sequence of deforming operations. It allows various granularities of deformation. This matches the rough-to-fine design process and allows hierarchical deformation. All these capabilities make the technique very useful in interactive 3D surface modeling.

Fig. 4 Surface S_1.

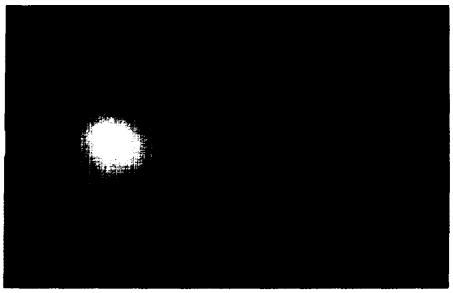

Fig. 5 Deformation surface S_2.

Fig. 6 Surface to be deformed S_3.

Fig. 7 Deformed surface S_4.

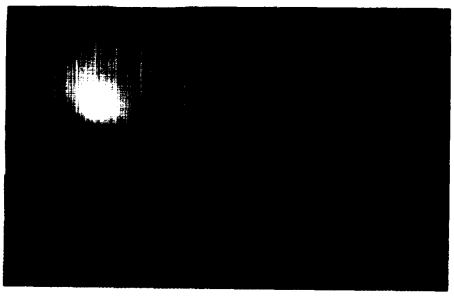

Fig. 8 Deformed surface S_5.

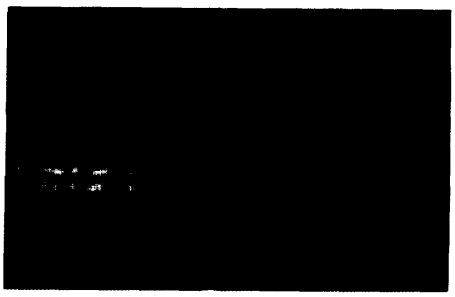

Fig. 9 Surface to be deformed.

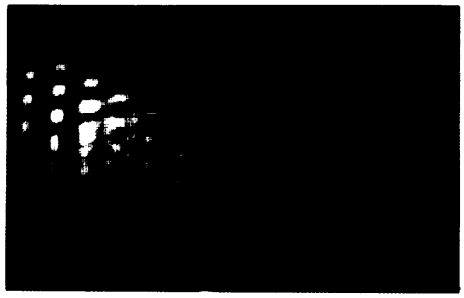

Fig. 10 Deformed surface.

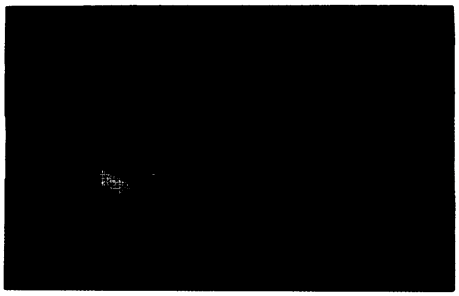

Fig. 11 Surface to be deformed.

Fig. 12 Deformed surface.

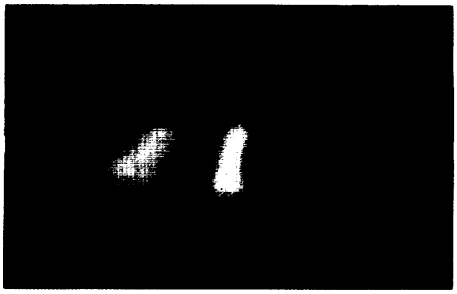

Fig. 13 Deformation surface.

Fig. 14 Deformed surface.

Fig. 15 Combination of deformations.

References

1. BARR, A. H. Global and Local Deformations of Solid Primitives, *Computer Graphics*, **18**, 3 (July 1984), 21-30.
2. BARSKY, B. A. and BARTELS, R. A. *An Introduction to Splines for Use in Computer Graphics & Geometric Modeling*, Morgan Kaufmann Publishers, Inc. (1986).
3. BARTELS, R. and HARDOCK, R. T. Curve-to-Curve Associations in Spline-Based Inbetweening and Sweeping, *Computer Graphics*, **13**, 3 (July 1989), 167-174.
4. BLOOMENTHAL, J. and SHOEMAKE, K. Convolution Surfaces, *Computer Graphics*, **25**, 4 (July 1991), 21-30.
5. BROOKS, F. P. JR. Grasping Reality Through Illusion Interactive Graphics Serving Science, *Proceedings of SIGCHI'88 Conference on Human Factors in Computing Systems* (May 1988).
6. COHEN, E., LYCHE, T. and RIESENFELD, R. Discrete B-Splines and Subdivision Techniques in Computer-Aided Geometric Design and Computer Graphics, *Computer Graphics and Image Processing*, **14**, 2 (1980), 87-111.
7. COQUILLART, S. Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling, *Computer Graphics*, **24**, 4 (Aug. 1990), 187-196.
8. COQUILLART, S. and JANEÉNE, P. Animated Free-Form Deformation: An Interactive Animation Technique, *Computer Graphics*, **25**, 4 (Aug. 1991), 23-26.
9. EMMETT, A. Guardians of Style, *Computer Graphics World* (June 1991), 30-40.
10. FARIN, G. *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, Inc., Second Edition, (1990).
11. FAUX, I. D. and PRATT, M. J. *Computational Geometry for Design and Manufacture*, Ellis Horwood Ltd. (1979).
12. FOLEY, J. D., VAN DAM, A., FEINER, S. K. and HUGHES, J. F. *Computer Graphics: Principles and Practice*, Addison-Wesley Publishing Company, Inc., Second Edition (1990).
13. FORSEY, D. R. and BARTELS, R. H. Hierarchical B-Spline Refinement, *Computer Graphics*, **22**, 4 (Aug. 1988), 205-212.
14. JERARD, R. B., DRYSDALE, R. L., HAUCK, K., SCHAUDT, B. and MAGEWICK, J. Methods for Detecting Errors in Numerically Controlled Machining of Sculptured Surfaces, *Computer Graphics and Applications*, **9**, 1 (Jan. 1989), 26-39.
15. PARENT, R. E. A System for Sculpting 3-D Data, *Computer Graphics*, **12**, 2 (July 1977), 138-147
16. PLATT, J. C. and BARR, A. H. Constraint Methods for Flexible Models, *Computer Graphics*, **22**, 4 (Aug. 1988), 279-288.
17. SCLAROFF, S. and PENTLAND, A. Generalized Implicit Functions For Computer Graphics, *Computer Graphics*, **25**, 4 (July 1991), 247-250.
18. SEDERBERG, T. W. Free-Form Deformation of Solid Geometric Models, *Computer Graphics*, **20**, 4 (Aug. 1986), 151-160.
19. TERZOPOULOS, D., PLATT, J., BARR, A. and FLEISCHER, K. Elastically Deformable Models, *Computer Graphics*, **21**, 4 (July 1988), 205-214.
20. TERZOPOULOS, D. and FLEISCHER, K. Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture, *Computer Graphics*, **22**, 4 (Aug. 1988), 269-278.

(Received October 4, 1991; revised February 20, 1992)