PARAMETRIC SURFACES WITH DISPLACEMENT AND DEFORMATION

Miguel CHOVER
Jaume I University
Castellón, SPAIN

José RIBELLES
Jaume I University
Castellón, SPAIN

Emilio CAMAHORT
University of Texas
Austin, USA

Joaquín HUERTA
Jaume I University
Castellón, SPAIN

Ricardo QUIRÓS
Jaume I University
Castellón, SPAIN

ABSTRACT

Parametric surfaces are widely used for modeling in CAD systems. However, they pose problems when representing complex macrogeometric features. In order to solve these problems we introduce an extension to parametric surface models that incorporates displacements and deformations. Our extended model retains its original parametric properties, thus allowing easy texture mapping together with a better macrogeometric representation. We allow both two-dimensional and three-dimensional functions to generate displacements that can be applied along directions different from the normal, allowing a continuous level of detail variation, as a function of distance to the viewer. Additionally, we compute deformations using geometric transformations that produce global effects on an entire object. We present object-space algorithms that allow direct rendering of the parametric surfaces without conversion to polygonal approximation, and smooth transitions between bump rendering algorithms. Our paper includes applications of our method to tree modeling and ceramic tile design. Key words: geometric modeling, surface detail, displacement functions, deformation techniques.

1. INTRODUCTION

Macrogeometric surface features modeling is an interesting field of research whose origin is closely related to texture mapping techniques. The first attempt to generate bumps over a surface was proposed by James F. Blinn (1978). His method, named bump mapping, produces tiny bumps relatively small with respect to the surface size. Such bumps do not occlude each other, and can not be seen when looking along the side of the surface.

To solve these problems, Robert L. Cook (1984) introduced the idea of displacement mapping. Displacement mapping construct a grid of micropolygons from a given surface. Then, the micropolygons are displaced, so those surface features become apparent. Cook's approach requires high-resolution grids for proper modeling of surface detail. Such grids require, in turn, large data structures and computationally expensive algorithms to guarantee the quality of results. J. W. Patterson *et al.* (1991) introduced an inverse displacement technique that generates models without increasing the amount of geometry. J. R. Logie and J. W. Patterson (1995) later generalized that technique. Their method requires a lot of computation, and, still, applies only to a limited class of surfaces. Furthermore, they do not provide a means of generating the new improved geometry, as required by CAD/CAM systems. All of the above methods use two-dimensional functions to model surface detail. Alternatively, Ken Perlin (1989), and Bradley A. Payne and Arthur W. Toga (1992) proposed the use of three-dimensional functions that are represented as density volumes instead of surfaces. Therefore, they require expensive algorithms to render these volume functions. Common displacement mapping techniques move surface points along the direction of the surface's normal. Hans Køehling Pedersen (1994) improved on these techniques by displacing surface points along trajectories defined by flow directions (flow mapping). Some of our work is related to specifying displacement directions different from the surface normal.

In addition to displacement, we also generate irregular surface detail by deforming the objects themselves. The two most important deformation methods, which are independent of the underlying representation of an object, were presented in (Bechmann, 1994). The first one, due to Alan H. Barr (1994), generalizes standard modeling operations to obtain non-linear global deformations. The second one, due to T. W. Sederburg and S. Parry (1986), obtains object deformations by perturbing the space in which the object is immersed.

In our work we improve upon these methods by providing a more powerful model to represent macrogeometric features for parametric surfaces. Specifically, we introduce a set of displacement and deformation functions that apply to a base parametric surface. Such functions are based on both two and three-dimensional texture functions. Keeping the parametric surface representation at all times allows us to add surface detail by applying textures. We render our model using a modified version of Edwin Catmull's curved surface rendering algorithm. Our version is more efficient and allows smooth transitions between different bump rendering algorithms. This paper concludes with some applications of our method.

2. SURFACE DEFINITION

Given a base surface S, we define a new displaced and deformed surface by specifying a displacement function F and a global deformation matrix M. The function F models a displacement along the direction of the surface normal N.

$$\mathrm{S}'(u,v) = \mathrm{M} \cdot \left(\mathrm{S}(u,v) + \mathrm{F} \cdot \mathrm{N}\right)$$

For the purpose of our discussion and to avoid multiplications by zero, we express matrix M as its upper-left 3x3 rotation-and-scaling submatrix R, and its upper-right 3x1 translation column vector T. Given that representation, we reformulate our model's equation as:

$$\mathrm{S}'(u,v) = \mathrm{R} \cdot \left(\mathrm{S}(u,v) + \mathrm{F} \cdot \mathrm{N}\right) + \mathrm{T} \qquad \mathrm{M} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where:

- S is a function representing the parametric surface.
- F is a function that gives three magnitude components for the displacement along the given direction. F can be a two-dimensional (displacement map) or a three-dimensional function.
- N represents the displacement direction. Our model allows N to depend on any parametric surface. N may also be chosen from a 3D vector field.
- R is the rotation and scaling matrix defined above. The elements $r_{i,j}$ of R may be functions, usually representing a non-linear geometric transformation.
- T is a translation vector, whose components may also be defined as functions.

We can now use this surface equation to obtain tangent vectors along the parametric directions. This can be easily done by taking derivatives with respect to the parametric variables u and v. Since displacement and deformation parameters may be expressed as functions of those variables, taking derivatives can be cumbersome. Instead, we sample the surface function inside a small interval in each direction u and v. The tangent vectors can then be expressed as:

$$\mathrm{S}'_u(u,v) = \frac{\left(\mathrm{S}'(u+\varepsilon,v) - \mathrm{S}'(u-\varepsilon,v)\right)}{2\cdot\varepsilon}$$

$$\mathrm{S}'_v(u,v) = \frac{\left(\mathrm{S}'(u,v+\varepsilon) - \mathrm{S}'(u,v-\varepsilon)\right)}{2\cdot\varepsilon}$$

We then obtain the normal to the surface by computing the cross product of these two tangent vectors.

This representation allows us to precisely define the new irregular surface given the different components of our model. Using the algorithms described in section 5 can then directly render the new surface. This is an alternative to common practice that resorts to polygonal approximations for rendering surfaces of this type. So, our model is more flexible, since it allows both direct and approximate rendering. This is useful for certain CAD/CAM applications that require the original surface models rather than a polygonal approximation.

3. DISPLACEMENT FUNCTIONS

We use two functions to express displacement from the base surface S. Function F gives the magnitudes of the displacement, and function N gives the direction of the displacement. Both can be functions of two or three variables, and both return a 3D vector. We require both functions to be smooth in order to avoid cracks and holes.

3.1 Displacement magnitudes.

In order to provide level-of-detail control to our rendering algorithm, we allow F to depend on the distance to the viewer. That way we guarantee that only the necessary level of detail will be generated for rendering. We also allow the definition of different functions for each of the X, Y, and Z components, so as to provide a more flexible means for representing surface detail. We distinguish two types of displacement magnitude functions.

- Functions of two variables F(u,v). These functions depend on the parameter variables u and v. We can define a unique function for all three X, Y, and Z components, or one function for each component. Such functions have traditionally been defined as arrays of displacement values. We support a mip-mapped representation of these arrays that allows level-of-detail control at rendering time. Level of detail is specified by a continuous parameter d computed by linear interpolation between the nearest two levels. u and v are also continuous parameters computed using bilinear interpolation.
- Functions of three variables F(x,y,z). Functions of three variables take as input a 3D point, and return either a scalar or a vector. For these functions we use modified versions of the Noise and Turbulence functions introduced by Perlin (1985). A displacement magnitude is obtained by evaluating a function at a 3D point on the original surface S.

Let P be one of those points with coordinates (x,y,z). Then, the turbulence function is defined as:

$$Turbulence(\mathrm{P}) = \sum_{i=0}^{k} abs\left(\frac{Noise(2^i \cdot \mathrm{P})}{2^i}\right)$$

In order to provide level-of-detail control to this function we add a parameter d in [0..1] that interpolates linearly between two consecutive levels of detail:

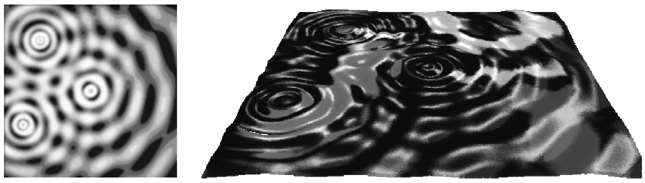


Figure 1. Example of displacement function applications along the normal to the surface

3.2 Displacement Direction.

Traditionally displacement has been defined along the normal to the base surface. This is a limitation that restricts the simulation of certain effects (Køhling, 1994). To avoid this limitation we allow functions that generate displacements in different directions.

- Normal to a Parametric Surface. We use the normal to a parametric surface S_i defined separately from the base surface S. We then used that normal to determine the displacement on surface S, given the magnitudes obtained by evaluating F.

$$S'(u,v) = S(u,v) + F \cdot N(S_i(u,v))$$

- Vector Spaces. We use solid textures to generate the displacement directions. Specifically, we use functions based on the gradient of Perlin's Noise function. Such functions are parameterized by a 3D point (x,y,z), and return the normal vector at that point.

$$S'(u,v) = S(u,v) + F \cdot DNoise(S(u,v))$$

Figure 2 shows an example of how to use these functions to obtain displacements along directions different from the normal to the surface.

Figure 2. Displacement along directions given by a 3D vector field

4. DEFORMATION MODEL

Our deformation model is based on Barr's model (Barr, 1984). Barr's model is independent of the underlying representation of an object. So, we can easily adapt it to handle parametric surfaces. Furthermore, we add new parameters that vary in a non-linear fashion as a function of a 3D solid texture (Perlin, 1985). Following Barr's notation, we define a standard translation as:

$$(X,Y,Z) = (x + t_x, y + t_y, z + t_z)$$

Now we can use the Noise function to compute non-linear global deformations with random behavior:

$$(X,Y,Z) = (x + k \cdot Noise(P), y + k \cdot Noise(P), z)$$

where k is a Noise scale factor and P a surface point before the deformation. A transformation matrix can specify simple deformation functions. More complex deformations can be obtained by composing different transformation matrices. The resulting matrix M, is the one we introduced in section 2. We have used some of these deformation functions to produce effects on trees and other plants. These effects include irregularities and tropisms (due to wind, gravity, etc.) like those depicted in figure 3.

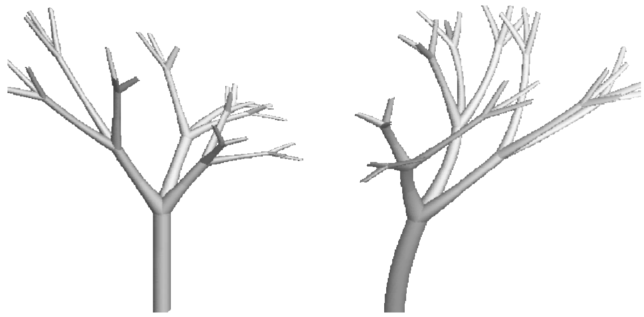


Figure 3. Effects of deformation functions

5. VISUALIZATION

Recent rendering techniques for parametric surfaces are based on incremental algorithms for bicubic patches (Rappoport, 1991) (Klasen, 1994). The complexity of our model, however, requires a different and more general approach. Specifically, we want an algorithm suited for any type of parametric surface, even non-polynomial ones. Also, we want to avoid expensive intersection computations in object space. It turns out that Catmull's recursive subdivision algorithm provides most of these features. The only problem with this algorithm is that subpatch projections are recomputed every time a patch is subdivided. This is a wasteful approach, since subpatch coordinates projections are shared among them and with the original patch. One solution for this problem precomputes the coordinates of the patch projections before starting the recursive subdivision. We can also render surfaces with shadows. Shadows are computed using Williams' algorithm (Williams, 1978), which is suited for any object that can be rasterized, including curved surfaces. Its only problem is aliasing, a problem common to other scan-conversion algorithms. Some solutions have been proposed to solve this problem (Reeves, 1987).

6. SMOOTH TRANSITION RENDERING

Bumped surfaces can be rendered using both displacement and normal perturbation algorithms. For objects closer to the viewer displacement techniques provides better results at higher computational expense. Conversely, objects further from the viewer can still be rendered using normal perturbation without noticeably reducing the image's quality. It is obvious then that a hybrid approach is most desirable. In order to incorporate both algorithms into our system we had to take into account two problems: how to classify the patches, and how to smooth out transitions between rendering algorithms.

We have developed an approach based on work by Becker and Max (1993), but we do not represent microgeometric surface properties as they do. We do consider 3D normal perturbation functions compatible with displacement mapping functions, always according to their approach (Max, 1994).

The idea of the algorithm is to progressively reduce the displacement of a surface, as it moves away from the viewer. At some point the displacement does no longer have to be computed, and can instead be simulated by a normal perturbation. We can then classify patches into three categories, according to their distance to the viewer:

- Close patches are rendered using displacement method. As the patch moves away from the viewer, its surface displacement is reduced linearly. Normals to the surface are computed geometrically after applying the displacement function.
- Intermediate patches are also rendered using a displacement method, but their normals are obtained by linear interpolation between the displaced surface normals and the perturbed normals.
- Far patches are rendered without displacement, but just normal perturbation.

The table 1 summarizes running times of this algorithm when applied to the scenes depicted in figure 4. Tests were run with both 2D and 3D displacement functions on a Series 715 Hewlett-packard workstation.

Normal Perturbation

Smooth Transition

Displacement

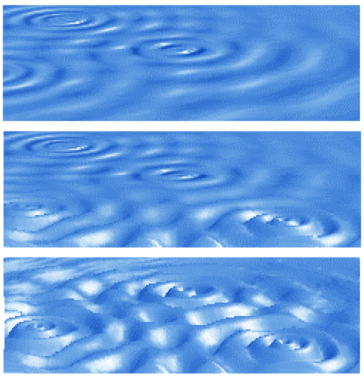


Figure. 4. A comparison between different bump rendering algorithms

	Normal Perturbation	Smooth Transition	Displace.
2D Displace.	10,4s.	24,0s.	37,5s
3D Displace.	12,2s.	28,3s.	44,3s.

Table 1. A comparison between running times

7. APPLICATIONS

Displaced and deformed surfaces allow the representation of objects that are otherwise difficult to model using conventional modeling techniques. We have successfully applied our models to the representation of trees and ceramic tiling patterns (Chover, 1995) (Chover, 1996).

7.1 Representation of Trees.

In order to represent trees we first obtain a skeleton of a tree by deriving the geometry of its branches using an L-system. Once the skeleton has been generated, we model the branches using truncated cones. Finally, we add geometric detail to the branches by providing different levels of displacement and deformation. Traditional texturing techniques can also be used for increased color and illumination realism.

Branches are displaced using functions of two and three variables that allow different branching effects, and an accurate representation of the tree's bark. We can also add other effects, like snow, by slightly displacing upwards the surfaces of the object whose orientation is up. Note that such surfaces need to be properly textured after displacement in order to obtain the right color at rendering time (see figure 5). Additionally, we can deform the entire tree to obtain more spectacular effects, like those produced by wind and the influence of gravity.

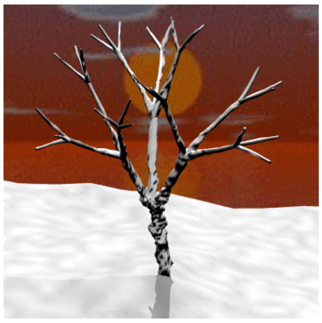
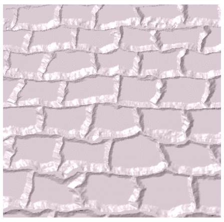


Figure 5. Tree rendering

7.2 Representation of Ceramic Tiling.

We have successfully applied our technique to generating artistic surface patterns for tile manufacturing. We can automatically produce surfaces with relieve patterns given a specification in our model. We then polygonalize the surfaces and feed the data into a numeric control machine. Our work also involves helping with the design of appropriate relieve patterns (see figure 6).



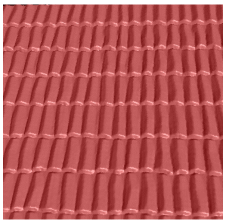


Figure 6. Displacement Functions

8. CONCLUSIONS AND FUTURE WORK

Our method makes several contributions to the generation of macrogeometric features for parametric surfaces. First, we allow functions of both two and three variables to compute the parameters of the representation. Second, we incorporate displacement functions that apply to directions different from the surface normal. Third, we provide level-of-detail control for the above displacement. Finally, we derive a method to deform displaced models based on Barr's deformation techniques.

Our results also include an algorithm for direct rendering of our surfaces that does not require polygonal approximation. Its main advantage is that it avoids the costly intersection computations that inverse approximation algorithms need (Logie, 1995). Also, we provide a means to achieve smooth transitions between different bump rendering algorithms.

We plan on continue our work in this area by devising new displacement and deformation functions, different from those already surveyed in this paper. We also plan on improving the efficiency of our algorithms, especially our rendering algorithm. Our final goal is to obtain a powerful and comprehensive model capable of modeling a wide variety of surface features.

ACKNOWLEDGMENT

The Fundació Caixa Castelló-Bancaixa supports this work.

REFERENCES

Barr, A.H., 1984, "Global and Local Deformations of Solid Primitives", *Computer Graphics*. Vol. 18, No. 3, pp. 21-30.

Becker, B.G. and Max, N. L., 1993, "Smooth Transitions between Bump Rendering Algorithms", *Computer Graphics*, pp. 183-190.

Bechmann, D., 1994, "Space Deformation Models Survey", *Computer & Graphics*, Vol. 18, No. 4, pp. 571-586.

Blinn, J. F., 1978, "Simulation of Wrinkled Surfaces", *Computer Graphics*, pp. 286-292.

Cook, R.L., 1984, "Shade trees", *Computer Graphics*, Vol. 18, pp. 223-231.

Chover, M., Vivó, R., Quirós, R. and Lluch, J., 1995, "Texture, Displacement and Immersion: A Model for Tree Rendering", *The Winter School of Computer Graphics and Visualization*, pp. 69-78.

Chover, M., Vivó, R., Quirós, R. and Lluch, J., 1996, "Modeling Urban Environments using Texture and Displacement Maps", *Spring Conference on Computer Graphics*, pp. 231-238.

Klasen, R.V., 1994, "Exact Integer Hybrid Subdivision and Forward Differencing of Cubics", *ACM Transactions on Graphics*, Vol. 13, No. 3, pp. 240-255.

Pedersen, H.K., 1994, "Displacement Mapping Using Flow Fields", *Computer Graphics*, pp. 279-286.

Logie, J.R. and Patterson, J. W., 1995, "Inverse Displacement Mapping in the General Case", *Computer Graphics Forum*, Vol. 14, No. 5, pp. 261-273.

Max, N. and Becker, B., 1994, "Bump Shading for Volume Textures", *IEEE Computer Graphics and Applications*, pp. 18-20.

Patterson, J.W., Hoggar, S.G. and Logie J.R., 1991, "Inverse Displacement Mapping", *Computer Graphics Forum*, Vol. 10, No. 2, pp. 129-139.

Payne, B.A., and Toga A.W., 1992, "Distance Field Manipulation of Surface Models", I*EEE Computer Graphics and Applications*, pp. 65-71.

Perlin, K., "An Image Synthesizer", *Computer Graphics*, Vol. 19, No. 3, pp. 287-296.

Perlin, K. and Hoffert, E.M, "Hypertexture", *Computer Graphics*, Vol. 23, No. 3, pp. 253-262.

Rappoport, A., 1991, "Rendering Curves and Surfaces with Hybrid Subdivision and Forward Differencing", *ACM Transactions on Graphics*, Vol. 10, No. 4, pp. 323-341.

Reeves, W., 1987, "Rendering Antialiased Shadows with Depth Maps", *Computer Graphics*, Vol. 21, No. 4, pp. 283-291.

Sederberg, T. and Parry, S., 1986, "Free-Form Deformation of Solid Geometric Models", *Computer Graphics*, Vol. 20, No. 4, pp. 151-160.

Williams, L., 1978, "Casting curved shadows on curved surfaces", *Computer Graphics*, Vol. 12, No. 3, pp. 270.

ABOUT THE AUTHORS

Miguel Chover Sellés, Ph.D, is a Professor in the Department of Computer Science, Jaume I University (SPAIN). He can be reached by e-mail: chover@inf.uji.es