Interactive Face-Replacements for Modeling Detailed Shapes

Eric Landreneau
Computer Science Department

Ergun Akleman*
Visualization Sciences Program
Department of Architecture

John Keyser
Computer Science Department

Texas A&M University

Abstract

In this paper, we present a method that allows novice users to interactively create partially self-similar manifold surfaces without relying on shape grammars or fractal methods. Moreover, the surfaces created using our method are connected. The modelers that are based on traditional fractal methods or shape grammars usually create disconnected surfaces and restrict the creative freedom of users. In most cases, the shapes are defined by hard-coded schemes that provide only a few parameters that can be adjusted by the users. We present a new approach for modeling such shapes. With this approach, novice users can interactively create a variety of unusual and interesting partially self-similar manifold surfaces.

1. Motivation

There exists a strong interest in contemporary sculpture and architecture to extend the limits of conceptual design by using rule-based generative systems. Several architectural design studios experiment with rule-based approaches. Designers in these studios use a wide variety of rule-based techniques such as L-systems, or cellular automata.

For designers, it is important to easily develop new generative procedures to have a variety of alternatives. However, for them it is hard to identify the rules. Shape construction with rules-based methods is generally a rigid and unpredictable process. Users input formulae with little or no interactive control over the resulting shape. Manipulating the end result is often costly and difficult.

* Corresponding Author: Visualization Sciences Program, Department of Architecture. Address: 216 Langford Center, College Station, Texas 77843-3137. email: ergun@viz.tamu.edu. phone: +(979) 845-6599. fax: +(979) 845-4491.

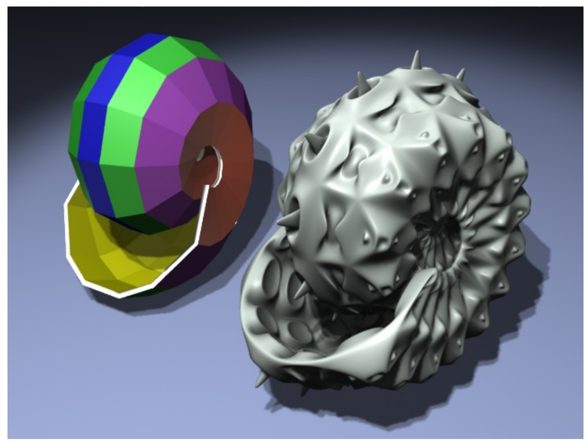


Figure 1. Using our face coloring and group extrusion methods, users can add finer details with greater control. (See Appendix for detailed explanation of the interactive modeling process)

Our goal in this paper is to blend rule based techniques such as fractals or L-systems with increased interactivity for designers. Although our motivation comes from fractals and L-systems, we want to give designers interactive control of modifiers to achieve conceptual shapes. We also want the resulting shapes to be physically constructible using 3D printers.

In this paper we present a simple approach that allows interactive and 3D extensions to Fractal and L-system methods. Using this approach, we have developed a system that enables designers to control each step of the shape generation process with a high level of interactivity.

With the new approach, novice users can easily create a large set of connected ”self-similar” manifold surfaces. Disconnected surfaces are acceptable for ”virtual” computer graphics applications in which the objects are used for only display purposes. However, in Architecture we usually want to physically construct the resulting shapes. To be able to construct the shapes, the shapes need to be connected and manifold surfaces. Figure 1 shows one example of how users can add finer details with our method.

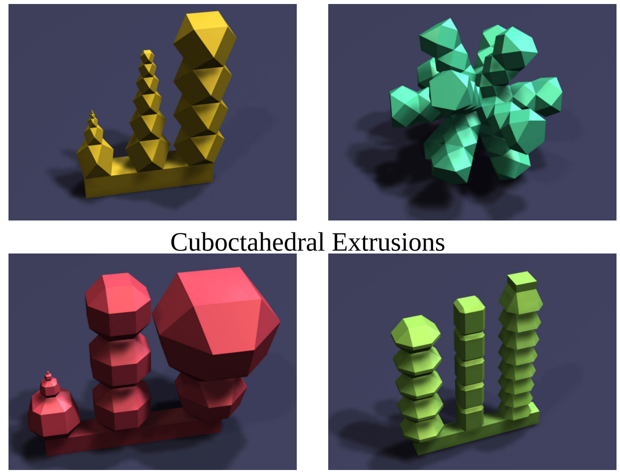
Disconnected manifold surfaces (if individual surfaces are manifold) can be printed but it is not possible to guarantee that the resulting physical object will stay together. If individual surfaces are not manifold they will not even be suitable for 3D printing. For instance, two methods based on Iterated Function Systems (IFS) [6] create a set of disconnected points or shapes, which can never be printed. In contrast, our method allows us to construct connected manifold surfaces which can be realized a 3D printer.

Our approach is based on face replacements, which are a generalization of the line replacements of 2D fractal geometry. Face replacements are created by using local mesh operators. These operators can be applied to one face of the mesh without affecting the rest. They replace the face with multiple faces. A local operator can be defined by a set of insert edge and create vertex operations [1, 3]. The local mesh operations such as extrusions guarantee that the resulting shape continues to be connected and manifold.

Landreneau et al. recently introduced Platonic extrusions as local mesh operators [14]. These extrusions, except tetrahedral extrusion, are generalized pipes in which bottom and top polygons have the same number of sides [14]. For this paper, we have extended Platonic extrusions to certain Archimedean extrusions as shown in Figure 2. Having a large variety of extrusions provides novice users a simple way to make face replacements. In addition to using these general extrusions, we introduce four new concepts for interactive modeling of connected and self-similar manifold surfaces: (1) *Face Grouping* using colors; (see Figure 1)(2) *Group extrusions*; (3) *Automatic Face Regrouping* and (4) *Remeshing Schemes*. Our method based on these concepts is guaranteed to create connected manifold surfaces.

This modeling approach moves towards a more hands-on approach to grammar based surface modeling. A user can assert much more control over the surface beyond the traditional *plug-in-a-formula-and-wait* method of generating grammar based models. Our approach will particularly be useful in Architectural concept modeling, in which users can quickly determine the effects of various approaches by simply recoloring the faces.

With this approach, users can rapidly learn how to create a wide variety of polygonal meshes that resemble grammar based shapes. Figure 3 shows some examples of fractal-like polygonal meshes that were created by some of our students using the system, as one of the biweekly assignments of a geometric modeling class. The approach is not only limited to fractal looking shapes, however, it can also be used for creating a wide variety of shapes, such as the simple cityscape shown in Figure 4.



Rhombicuboctahedral Extrusions

Figure 2. Examples of local mesh operators.

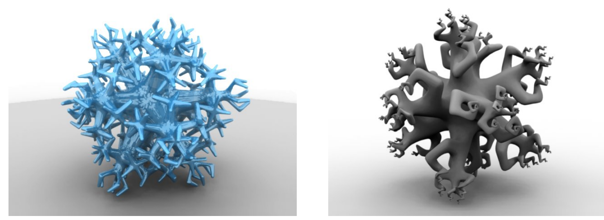


Figure 3. Examples of student works.

2. Previous Work

We observe that all the methods that allow creation of fine details are based on replacements. We will discuss previous work in three categories: Iterated Function Systems, Fractal Algorithms that depend on initial shape, and L-Systems.

2.1. Iterated Function Systems (IFS)

Fractal shapes most commonly are constructed with Iterated Function Systems, a simple procedure that exploits their self-similarity property, and introduced by Barnsley [6]. The Iterated Function Systems approach is based on the concept that self-similar shapes can be considered as a union of transformed (e.g. scaled, rotated, translated and mirrored) copies of itself. For instance, if a self-similar

Figure 4. A simple cityscape that is created by using group extrusions: Face grouping and group extrusions allow us to create similar looking but different buildings.

Fractal shape can be given as

$$S = \bigcup_{k=0}^{K} A_k S$$

where A_k is a 4×4 transformation matrix in homogenous coordinates. Barnsley introduced deterministic and probabilistic algorithms to construct such self-similar shapes.

- **Deterministic Algorithm:** This algorithm is based on shape replacements. A deterministic algorithm to construct a fractal shape involves creating a series of shapes starting from an initial shape S_0 as

  $$S_n = \bigcup_{k=0}^{K} A_k S_{n-1}$$

  In practice, it is possible to ignore the union operation since using disconnected copies will visually give the same results. In theory, a deterministic algorithm could compute actual union operations but this is seldom done in practice, due to the complexity of computation [21]. As a result, the number of the copies increases K times in each iteration. Since the number of copies increases very quickly, after a few iterations a good approximation of S can be obtained.

- **Probabilistic Algorithm:** In this case, the resulting shape is just a collection of points. We start with a randomly chosen initial point, and apply to this point a randomly chosen transformation A_k. This operation creates a new point. We again apply a randomly chosen operation to this point. After a few iterations, we start to record the positions of the points. The collection of all these points gives us the shape.

Because of their simplicity, these algorithms are widely used to create fractal shapes. Another important property of these algorithms is that they are dimension independent. The same conceptual algorithm can be used both for 2D and 3D shape construction. For a successful application of IFS in 3D, see the XenoDream software [22]. Moreover, these algorithms are independent of the way the shape is represented, e.g, S_0 can be a set of points, an implicit surface, a polygonal surface, or a NURBS surface. Regardless of the type of the initial shape, the same algorithm can be used.

One of the problems with these construction algorithms is that the algorithms are specific to the target (limit) shapes. Each algorithm approaches its target shape regardless of the shape of the initial object. These algorithms do not allow construction of different target shapes from different initial shapes.

The biggest problem with IFS based algorithms is that the constructed shapes are usually not connected, and thus cannot be printed in 3D.

2.2. Fractal Algorithms that Depend on Initial Shape

Fortunately, IFS is not the only method for constructing fractal shapes. There exist alternative approaches in which the resulting shape depends on initial shape. However, these alternative approaches are usually not dimension independent and are hard to implement in 3D; they have not been widely used in 3D applications.

2.2.1. Line Segments Replacements The line segments replacement method introduced by Mandelbrot is one such algorithm. Using line replacements, a wide variety of 2D fractals can be constructed [16], however, line replacements are useless in 3D.

For 3D, instead of line replacements we have to use face replacements. However, faces are not as simple entities as line segments. Line segments are always the same. They are straight and have a starting point and an ending point. After the replacement, one line segment is transformed into a set of line segments, which is exactly the same entity. Therefore, we can apply the algorithm iteratively.

It is not the same for faces. A face can have any number of corners and it may not be planar. Therefore, a face replacement method must be able to work on any type of face without affecting the rest of the mesh. Local mesh operations such as extrusions can exactly satisfy this criteria. They can be applied to any face and they do not affect the rest of the mesh.

For face replacements it is also important to create a similar version of the original face. Most extrusions also provide this property since they usually are generalized pipes in which the bottom and the top polygons have the same number of sides [2].

Although they cannot be applied locally, some subdivision algorithms such as the Doo-Sabin and Loop subdivision schemes [9, 15, 23, 20] can also be used as face replacements since they can provide smaller versions of faces of an initial mesh. Checkerboard subdivision, a scheme we have also introduced, provides the same property.

2.2.2. Set Difference There also exist approaches that allow construction of a variety of fractal shapes from different initial shapes. A notable example is one of Mandelbrot's alternative Sierpinski triangle constructions that relies upon "*cutting out 'tremas'*" as defined by Mandelbrot [16].

An attractive property of the such construction is that the initial shape does not have to be a uniform triangle (Mandelbrot did not explicitly mention it [16]). The initial shape can be a convex polygon by simply restating the construction algorithm as *from each convex polygon cut a convex polygon that is created by connecting the midpoints of each edge*. After the first iteration of this algorithm, all polygons become triangles. If we interpret the "cut" operation as an "exclusive-or" operation instead of a set-difference, we can also safely apply this construction to even non-convex shapes.

However, it is still hard to extend this algorithm to 3D using set operations. To construct a generalized Sierpinski polyhedron, we need to take a set-difference of the initial polyhedron with a polyhedron that is constructed by connecting midpoints of each edge in the original polyhedron. The problem is that the set difference operation needs to be implemented unlike union. Set difference can be particularly hard to implement since it can create non-manifold fractal shapes such as a Sierpinsky tetrahedron [12]. Construction of a polyhedron by connecting the midpoints of each edge of the initial polyhedron can also be hard in solid modeling. In the case of a tetrahedron, the problem is easy since the shape that is constructed is an octahedron; i.e. the faces are triangular and therefore planar. But for most cases, the faces may not be triangular and hence may not be planar, complicating the set-difference procedure even further.

A solution to these problems for Sierpinsky polyhedra has recently been provided by Akleman and Srinivasan using topological mesh modeling [21]. The problem with this approach is that it is useful only for creating a specific family of fractals, which are generalized, manifold, and connected versions of Sierpinsky tetrahedra.

2.3. L-Systems

L-systems are a much more powerful version of Fractal geometry's line segment replacement algorithms. In L-systems each line segment can have a label; based on the labels, each line can be replaced by different sets of lines. Because of their grammar based nature, L-systems can include context sensitivity and even parameters. These properties of L-systems make them very useful for designing the shapes of plants and trees.

Despite their power over simple line-replacements, L-systems suffer similar problems to those of line-replacements. 3D shapes from L-systems are created by replacing the original lines with surfaces (usually cylinders). Since its is hard to connect these surfaces, the shapes described by L-systems usually consist of disconnected pieces.

3. Methodology

Our goal in this paper is to achieve the grammar based power of L-systems for constructing connected and manifold surfaces by combining face replacements with face grouping. In this section, we discuss four concepts introduced together in this paper: (1) *Face Grouping* using colors; (2) *Group extrusions*; (3) *Automatic Face Regrouping* and (4) *Remeshing Schemes*. Our method based on these concepts is guaranteed to create connected manifold surfaces.

3.1. Face Grouping Using Colors

Face grouping using colors allows users to easily group the faces in any modeling software. In face grouping, users classify faces by assigning a color to each. Faces are classified by colors, with identically colored faces belonging to a common group. Note that this stage is completely under the user's control. The faces do not have to be geometrically or topologically similar, so there are no restrictions for assigning a color to a face.

3.2. Group extrusions

Group extrusions simplify multiple extrusions. The users can apply the same extrusion operation to all identically colored faces by selecting only one face (see Figure 5).

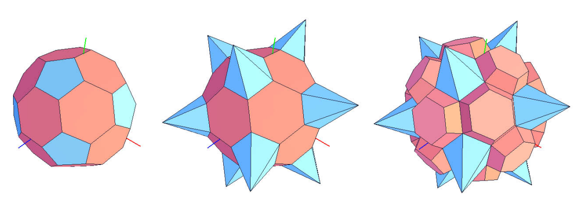


Figure 5. Face coloring and group extrusions.

Extrusions (except tetrahedral) produce a "top" face similar to the parent face, which is connected to the parent edges by "side" faces. The top face can inherent the group of the parent face. However, after a few iterations of remeshing or grouped extrusions, the number of side faces increases exponentially. We have provided automatic face regrouping to simplify the users' job regroup newly created side faces (see Figure 6).

3.3. Automatic Face Regrouping with Modulus Coloring

To regroup side faces, we introduce the modulus coloring concept. The side faces are regrouped according to a modulus scheme. Starting from a randomly chosen side face, new groups are generated using a user supplied modulus. With a modulus of 1, every side face would share the same group. A modulus of 2 would generate two alternating groups. A modulus of 3 will make every third side face the same group, and so on. Using a modulus equal to the number of side faces of the extrusion, equal to the number of edges in the parent face, will assign a unique group identity for each side face.

The modulus ensures that side faces will exhibit radial symmetry, due to side faces sharing colors. The modulus operation is not unique since the regrouping can be different based on the choices of initial side faces. Because of this, regrouping can introduce slight irregularities all allow to break overall symmetry of the object as seen in Figure 6.

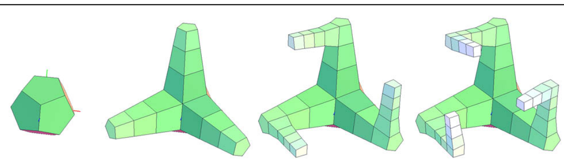
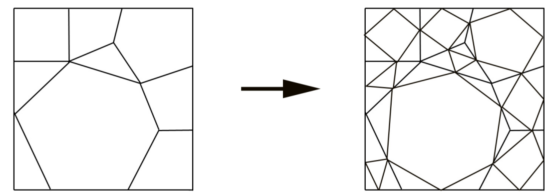


Figure 6. Automatic coloring newly created faces and group extrusions based on automatic coloring.

3.4. Remeshing Schemes with Group Extrusions

Based on the recent research on subdivision surfaces, there now exists a wide selection of remeshing schemes that can be used in interactive applications. It is possible to view these remeshing operations as face replacements that are applied in parallel. By combining these schemes with group extrusions we provide additional flexibility. In fact, Loop style remeshing is particularly common in fractal algorithms [11]. Using Loop style remeshing (see Figure 7), it is possible to create generalizations of Koch islands (see Figure 8) and Sierpinsky tetrahedra (see Figure 9). Loop style remeshing with random vertex displacements is widely used for terrain generation.



Initial mesh Resulting mesh

Figure 7. The remeshing scheme of Loop subdivision. Here, we first subdivide every edge at its mid-point and connect these mid-points to create a central face in the middle of each original face and triangles in each corner.

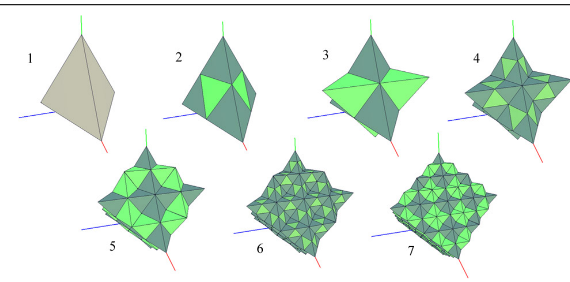


Figure 8. Loop remeshing scheme with tetrahedral extrusion. Using this procedure one can create Generalized 2-Manifold Koch Islands.

Existing subdivision remeshing algorithms can be categorized into two main classes [5]. These are conversion schemes and preservation schemes. Each of these can further be divided into primary and dual schemes.

- Primary Conversion schemes: These remeshing algorithms, which are called triangulization, quadrilateralization or pentagonalization algorithms, convert any given mesh to a triangular, quadrilateral and pentagonal mesh respectively. The remeshing scheme of $\sqrt{3}$ subdivision [13] is a triangulization algorithm. On the other hand, the remeshing scheme of Catmull-Clark subdivision [7], which is also called vertex insertion, is a quadrilateralization scheme. The dual of Simplest subdivision is also a quadrilateralization scheme [18]. There exists only one pentagonalization scheme [5].

- Dual Conversion schemes: The duals of primary conversion schemes give 3, 4 and 5-valent conversion schemes. For instance, the dual of vertex insertion, corner cutting [9], is a 4-valent conversion scheme. After one application of corner cutting the valence of each vertex becomes 4. Simplest is another 4-valent conversion scheme [18]. Honeycomb (Dual of $\sqrt{3}$), which has been discovered by three groups simultaneously [4, 8, 17], is a 3-valent conversion scheme that converts any manifold mesh to a mesh with only 3-valent vertices. There exists only one 5-valent conversion scheme, which is the dual of pentagonalization scheme [5].
- Primary Preservation schemes: These remeshing algorithms preserve a triangular, quadrilateral, pentagonal or even hexagonal property of the given mesh, e.g., if a mesh is already triangulated, a triangle preserving remeshing creates a triangular mesh. However, if the original mesh has some non-triangles, the resulting mesh does not become a triangular mesh. The remeshing algorithm of Loop subdivision is a good example of a preserving algorithm. Although Loop subdivision is only applied to triangular meshes, its remeshing scheme can be applied to any mesh as shown in Figure 7. We do not know of any quadrilateral preserving remeshing scheme among published subdivision schemes.
- Dual Preservation schemes: The duals of primary preservation schemes give 3-valent and 4-valent preservation schemes. An hexagonal preservation scheme, which is the dual of Loop subdivision has recently been discovered by two groups [10, 19, 17].

The most apparent reason behind the popularity of Loop style remeshing among Fractal algorithms is that Loop preserves initial faces in every iteration. This property is particularly useful for face replacements since some of the newly created faces inherit the properties of initial faces. However, Loop is not the only one that can provide this property. All dual conversion schemes such as Corner Cutting, Simplest or Honeycomb,and all preservation schemes (Loop belongs to this group) preserve initial faces in every iteration. They all can easily be used for interactive face replacement applications.

This classification is also helpful to identify missing remeshing schemes that can be useful for rule-based application. One such scheme we have identified and implemented is the so-called "checkerboard scheme". Checkerboard is a quadrilateral preservation scheme and it preserve initial faces. Checkerboard is not really a new scheme; it is used in Fractal geometry to create generalized Menger Sponges (see Figure10).

4. Implementation and Results

The concepts that is discussed in methodology section are implemented and included in our existing 2-manifold mesh modeling system [3]. Our system is implemented in C++ and OpenGL. All the examples in this paper were created using this system. Figures 11 and 12 show four examples created using the system.

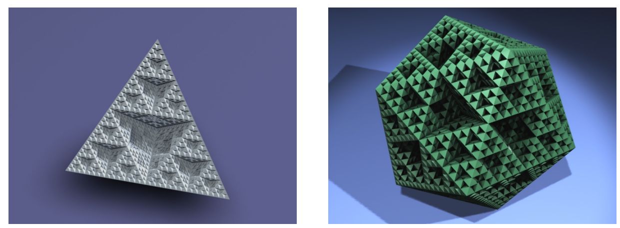


Figure 9. Two Sierpinsky tetrahedra look-a-likes that are created using our method. Since this method does not change the topology, we cannot have holes. On the other hand, the shape on the right is a generalized version that cannot be created by fractal methods. For generalized and truly high genus Sierpinsky polyhedra see [21]

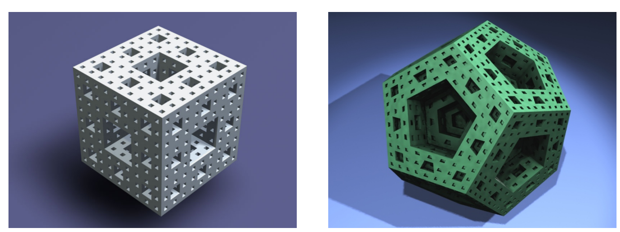


Figure 10. Two Menger Sponge look-a-likes that are created using our method. These are not real Menger sponges since using the method we cannot change the topology. On the other hand, The shape on the right is a generalized version that cannot be created by fractal methods.

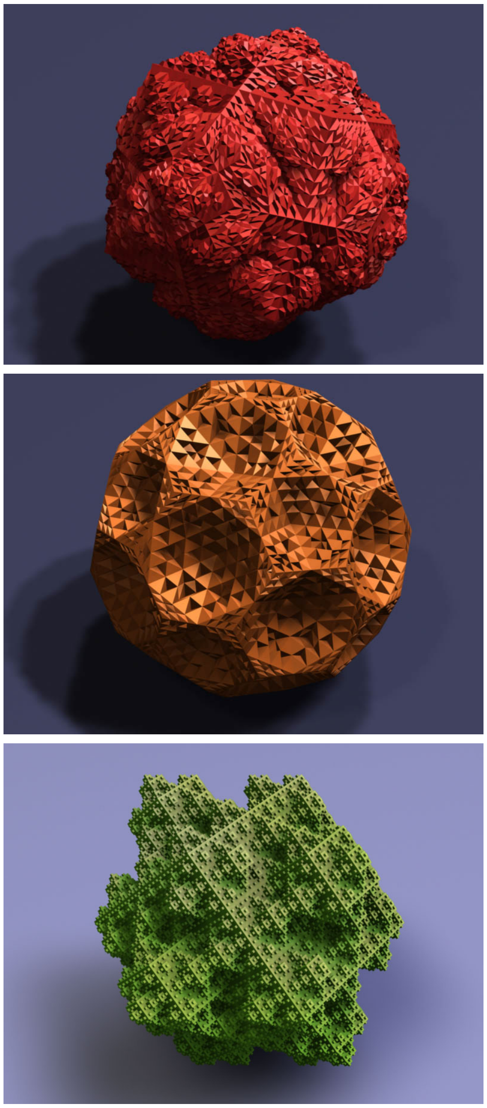


Figure 11. Interactively created fractal looking shapes.

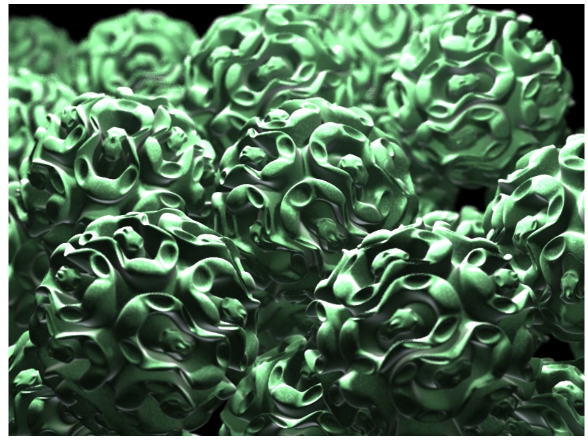


Figure 12. A detailed shape that is interactively constructed using our approach

5. Conclusion and Future Work

In this paper, we have presented an approach that allows designer to interactively create partially self-similar manifold surfaces without relying on shape grammars or fractal methods. Using this approach, we have developed a system that enables designers to control each step of the shape generation process with a high level of interactivity. With the new approach, designers can easily create a large set of connected "self-similar" manifold surfaces. Our method allows us to construct connected manifold surfaces which can be realized by a 3D printer.

References

[1] E. Akleman and J. Chen. Guaranteeing the 2-manifold property for meshes with doubly linked face list. *International Journal of Shape Modeling*, 5(2):149–177, 1999.

[2] E. Akleman and J. Chen. Regular meshes. In *Proceedings of Solid Modeling and Applications*, pages 213–219, 2005.

[3] E. Akleman, J. Chen, and V. Srinivasan. A minimal and complete set of operators for the development of robust manifold mesh modelers. *Graphical Models Journal, Special issue on International Conference on Shape Modeling and Applications 2002*, 65(2):286–304, 2003.

[4] E. Akleman and V. Srinivasan. Honeycomb subdivision. In *Proceedings of ISCIS'02, 17th International Symposium on Computer and Information Sciences*, volume 17, pages 137–141, November 2002.

[5] E. Akleman, V. Srinivasan, Z. Melek, and P. Edmundson. Semi-regular pentagonal subdivision. In *Proceedings of the International Conference on Shape Modeling and Applications*, pages 110–118, 2004.

[6] M. Barnsley. *Fractals Everywhere*. Academic Press, Inc. San Diego Ca., 1988.

[7] E. Catmull and J. Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer Aided Design*, (10):350–355, 1978.

[8] J. Claes, K. Beets, and F. V. Reeth. A corner-cutting scheme for hexagonal subdivision surfaces. In *Proceedings of Shape Modeling International'2002, Banff, Canada*, pages 13–17, 2002.

[9] D. Doo and M. Sabin. Behavior of recursive subdivision surfaces near extraordinary points. *Computer Aided Design*, (10):356–360, 1978.

[10] N. Dyn, D. Levin, and J. Simoens. Face-value subdivision schemes on triangulations by repeated averaging. In *Curve and Surface Fitting: Saint-Malo 2002*, pages 129–138, 2002.

[11] A. Fournier, D. Fussel, and L. Carpenter. Computer rendering of stochastic models. In *Proceedings of Computer Graphics, Siggraph*, pages 97–110, 1982.

[12] C. M. Hoffmann. *Geometric & Solid Modeling, An Introduction*. Morgan Kaufman Publishers, Inc., San Mateo, Ca., 1989.

[13] L. Kobbelt. $\sqrt{3}$-subdivision. In *Proceedings of SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 103–112. ACM, ACM Press / ACM SIGGRAPH, 2000.

[14] E. Landreneau, E. Akleman, and V. Srinivasan. Local mesh operations,. In *Proceedings of the International Conference on Shape Modeling and Applications*, pages 351 – 356, 2005.

[15] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[16] B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman and Co., New York, 1980.

[17] P. Oswald and P. Schröder. Composite primal/dual $\sqrt{3}$-subdivision schemes. Computer Aided Geometric Design, CAGD, 2003.

[18] J. Peters and U. Reif. The simplest subdivision scheme for smoothing polyhedra. *ACM Transactions on Graphics*, 16(4):420–431, 1997.

[19] H. Prautzsch and W. Boehm. Chapter: Box splines. The Hanbook of Computer Aided Geometric Design, 2000.

[20] M. Sabin. Subdivision: Tutorial notes. Shape Modeling International 2001, Tutorial, May 2000.

[21] V. Srinivasan and E. Akleman. Connected and manifold sierpinski polyhedra. In *Proceedings of Solid Modeling and Applications*, pages 261–266, 2004.

[22] G. Thornton and V. Sterling. Xenodream software,. http://www.xenodream.com, 2005.

[23] D. Zorin and e. P. Schröder. Subdivision for modeling and animation,. ACM SIGGRAPH 2000 Course #23 Notes, July 2000.

A. Modeling Nautilus Shell

The nautilus model that is shown in Figure 1 is colored according to stripes exist in shell. There is a main, central stripe, colored blue in Figure 1. On each side there is a green, purple, and red stripe. The lip of the model is colored white, and the inside of the model is colored yellow.

After coloration, grouped extrusions are performed. An extrusion performed on a stripe will apply that extrusion to every face on that stripe. In addition, it will also act on any other stripe of the same color. For example, performing a grouped extrusion on a green face will perform that extrusion on both green stripes.

The blue stripe has a series of three extrusions applied to it. An extrusion with a positive height, followed by a negative height, and another positive height extrusion is performed. This generates the spine shape seen along the middle of the object.

The green stripes take advantage of the modulus function in the extrusion. A negative height extrusion is performed with a modulus of two. Next, a side face is extruded. Due to the modulus function, this actually causes two faces to become extruded, as they share the same modulus. This generates the ”tooth” shapes corresponding to the green stripes on the original shape. A final extrusion is done on the middle of the shape, giving a bump in between the teeth.

The purple stripe is handled similarly to the green stripe. A two-modulus extrusion with a positive height is performed first. Next, a positive height extrusion is done on a side face. Finally, a negative height extrusion is performed on the middle face. This leads to the bumpy shapes corresponding to the purple stripes.

The red stripe involves a negative height extrusion with a two-modulus as well. A side face is then extruded.

The yellow interior faces receive a positive height extrusion followed by a negative height extrusion, creating sucker-like shapes. The white faces are left alone.

Finally, the shape is smoothed multiple times with Catmull-Clark remeshing.

These simple operations, while similar, generate a complex surface from the simple nautilus shell. Although complex, the shape exhibits features which correspond to the original coloring. In this case, there are patterns which follow the color stripes painted on the original model.