Constructing Models of the Seven-Around Surface

Mark Howison
Final Report
CS 285: Solid Modeling
Prof. Carlo Séquin

Introduction

The goal of this project was to construct computer models of a mathematical surface I have termed the “seven-around” surface, which has seven equilateral triangles adjacent to every interior vertex not on the boundary (see Figure 1). If there were instead six triangles at each vertex, the surface would be a planar tiling of hexagons; adding the seventh triangle causes the surface to buckle, taking on negative curvature. My approach was to design both a construction algorithm that stochastically adds faces to the surface and a control algorithm that determines via a rating system which models to continue constructing and which to abandon.

Mathematical Motivation

Mathematically, the seven-around surface is defined as the combinatorial manifold such that

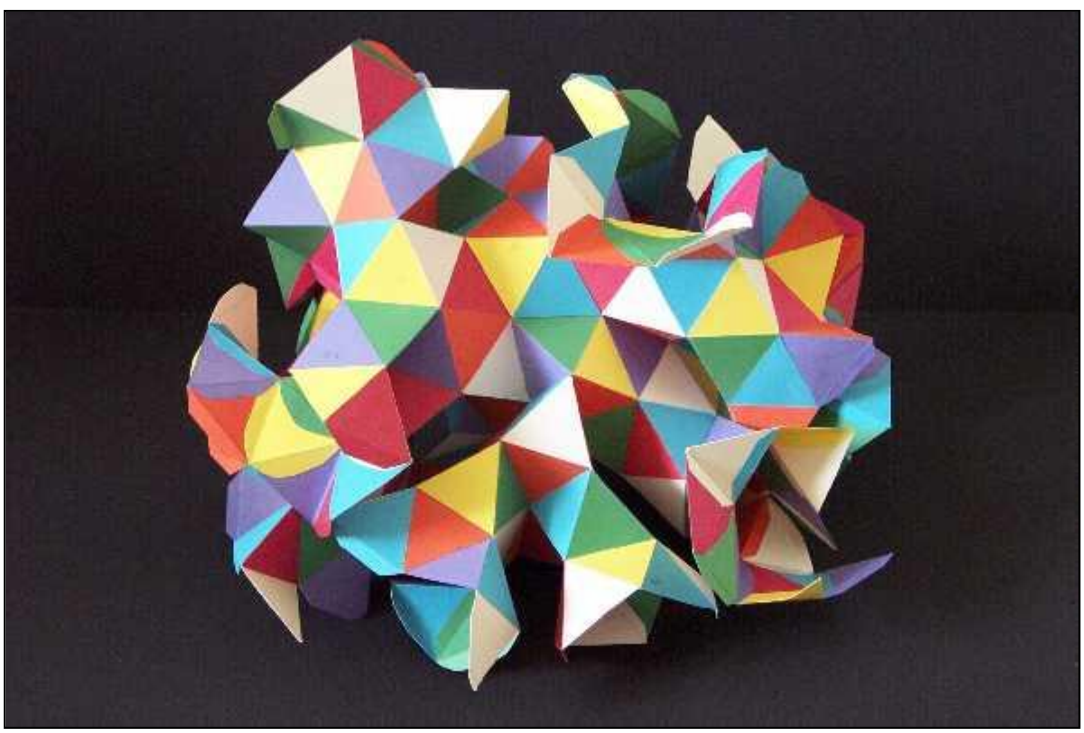

Figure 1. A paper model of the seven-around surface constructed by David A. Richter, Department of Mathematics, Western Michigan University. [http://homepages.wmich.edu/~drichter/hyperbolicimbed.htm]

the *star* (i.e. the set of faces adjacent to a given vertex) of every vertex contains exactly seven equilateral triangles. It is an open mathematical question as to whether this surface can be embedded in three-space, that is, constructed using infinitely many triangles without ever exhibiting a self-intersection (personal communications, Richard Schwartz and Thomas Banchoff). The models I have constructed are finite in size, and therefore have a boundary where the vertices do not satisfy the "seven-around" requirement.

One reason this surface is interesting is because it is *quasi-isometric* to the hyperbolic plane, meaning up to a certain, bounded error, distances measured along a given path on the seven-around surface are the same as those measured along an equivalent path in the hyperbolic plane. Because of this quasi-isomorphism, the seven-around surface is in some sense the combinatorial (i.e., discrete) analog to the smooth manifold of the hyperbolic plane. Hilbert proved that the hyperbolic plane cannot be smoothly and isometrically embedded in three-space. Therefore, it would be surprising if the seven-around surface can be embedded.

One intuition as to why it cannot be embedded is that the number of faces in a given bounding sphere increases exponentially with the radius of the sphere. Eventually, the faces need to be folded into parallel sheets. However, there is a lower bound on the distance between those sheets given by the size of the faces. This will lead to the formation of "air pockets," of which there will be exponentially many. This argument, however, is in no way a mathematically-rigorous proof. One potential problem with it is that many parallel sheets might share the same air pocket. In particular, it is unclear how to count or keep track of the air pockets (personal communication, Richard Schwartz).

The hyperbolic plane can be smoothly and isometrically embedded in $\mathbf{R}^6$, which is a result of D. Blanuša [1], although the embedding is not analytic.

Concentric Rings and the Control Algorithm

In the hyperbolic plane, the area of a circle grows exponentially as the radius of the circle increases. Analogously, if you choose an origin vertex on the seven-around surface and divide the rest of the surface into concentric rings centered at that origin, the number of faces in each ring increases exponentially. In fact, the sizes of the rings are given in Table 1.

Ring	Faces	Cumulative Faces	Color (see Figure 4)
0	7	7	Red
1	28	35	Orange
2	77	112	Yellow
3	210	322	Green
4	574	896	Blue
5	1568	2464	
6	4284	6748	
7	11704	18452	
8	31976	50428	

Table 1: The size of concentric rings in the seven-around surface.

This table of values can be calculated using a simple L-system. The key observations are the following:

1. Each adjacent pair of vertices on the boundary of ring *n* contributes a face to ring *n*+1.
2. After constructing the faces in (1), each boundary vertex of ring *n* has valence four or five. In other words, at each of these vertices either three or two faces, respectively, need to be added to ring *n*+1. Denote the first case by 3 and the second by 2.
3. Each 3 in ring *n*+1 propagates to a 3, 3, 2 in ring *n*+2. On the other hand, each 2 propagates to a 3, 3.
4. Each of the 2s and 3s in ring *n*+1 has two adjacent vertices on the boundary of ring *n*+1. Thus, the number of faces in (1) for ring *n*+2 is just the sum of the 2s and 3s in ring *n*+1.
5. The topology of the surface has seven-fold symmetry, so the above analysis only need be done for one seventh of the surface, and all the results multiplied by seven.

This leads to the following L-system for a seventh of the surface starting at ring 1:

Variables : 2 3
Constants : none
Start : 3
Rules : (3 → 3 3 2), (2 → 3 3)
Output :
(n=1) 3
$S(1) = 3$
(n=2) 3 3 2
$S(2) = 8$
(n=3) 3 3 2 3 3 2 3 3
$S(3) = 22$
(n=4) 3 3 2 3 3 2 3 3 3 3 2 3 3 2 3 3 3 3 2 3 3 2
$S(4) = 60$
. . .

S(*n*) denotes the sum of the L-system output at *n* and $S(0) = 1$. Using these summations, the expression for the number of faces in ring *n* is

$$7 \cdot [S(n) + S(n-1)].$$

Intuitively, it seems like there can't be enough "space" in three-space to accommodate this exponential growth. The exponential growth also causes problems for the construction algorithm. As *n* increases, a larger proportion of the vertices on the boundary of ring *n* are suffocated by all the other faces near them, and the construction algorithm can't figure out a non-intersecting way to insert the additional faces needed to move to ring *n*+1.

The basic idea of the control algorithm is to store completed *n*-rings and to give each one a rating based on how successful past expansion attempts for the ring were. Specifically, after an expansion attempt, the new rating R_{new} for an *n*-ring is given by

$$R_{new} = \frac{A \cdot R_{previous} + \dfrac{F - C(n)}{C(n+1) - C(n)}}{A+1}$$

where $R_{previous}$ is the rating before the expansion attempts, *A* is the previous number of expansion attempts, *F* is the number of faces in the expanded ring, and *C*(*n*) is the cumulative number of faces in an *n*-ring (see Table 1). The rating lies between 0 and 1, and can only be 1 if every expansion attempt has led to a new model with the entire next ring completed (0-rings will frequently have a rating of 1 since it is fairly easy to construct a 1-ring). The control algorithm preferentially uses the highest rated rings, but also continues to develop lower rated ones. Initially, during a "seed" phase it creates additional 0-rings (i.e., only the seven faces around the origin vertex) and expands rings at all values of *n*, but later in the "production" phase it uses only the highest and next-to-highest value of *n*, so as to conserve storage and processing time.

The Construction Algorithm

Each model has three stored data structures:

1. An array of vertices, stored as Vector3 objects (borrow from the OGRE open source graphics engine [2].
2. The star for each vertex, stored as a 7-tuple of vertex indices.
3. An array of faces, stored as 3-tuples of vertex indices.

These three data structures provide enough information for the algorithm to reconstruct the remaining data structures:

1. A map between face indices and a hash value for the indices of the 3 vertices incident to the face.
2. An array of the faces centers, again stored as Vector3 objects.
3. An array containing the minimum number of edge traverses back to the origin vertex, ordered by vertex index.
4. A queue of indices for vertices on the boundary of the largest completed ring, all of which have valence less than 7 (i.e., incomplete stars).
5. A stack of the indices for the boundary vertices whose stars could not be completed by the algorithm.

The algorithm starts by either loading in an existing model with *n* complete rings, or by starting with a single triangle and constructing a 0-ring. In the former case, the algorithm attempts to complete the star at each vertex on the boundary of the *n*-ring, which yields a (*n*+1)-ring if successful. Faces are added to the current star in two different ways, depending on whether there are any faces elsewhere in the model that are "intruding" into the bounding sphere around the current star.

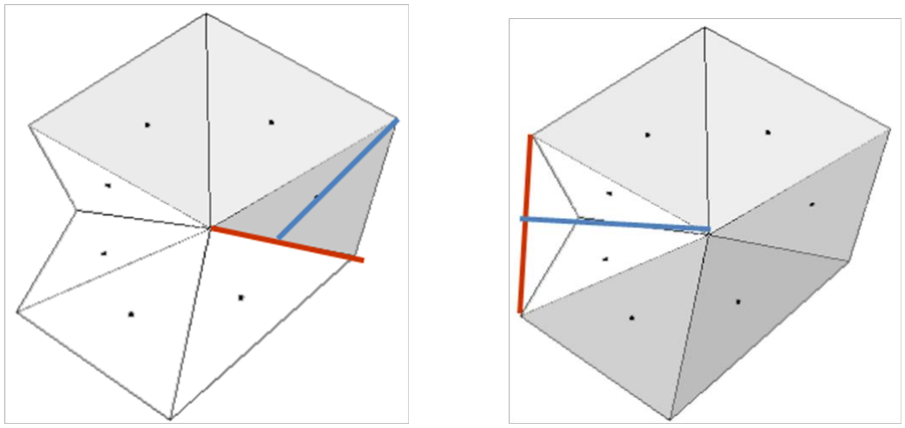

Figure 2. Illustrations of how the algorithm adds faces in the simple case when there are no faces intruding into the bounding sphere of the star. The gray faces are completed, and the white faces to-be-completed.

1. If there are no intruding faces, the previous face in the star is rotated around its leading edge (the orange line in Figure 2, left) by 180 degrees plus or minus an adjustment angle chosen between 0 and 90 degrees with uniform distribution. Also, the sign of the adjustment changes between successive faces, so that if the previous face had an added adjustment, the next face will have a subtracted one. Finally, the last two faces can be added only if the length of the segment between the first and last vertices of the current star (the orange line in Figure 2, right) is less than or equal to $\sqrt{3}$ (i.e., twice the height of a unit equilateral triangle). If this is true, the faces are restricted to two positions, either facing up or down, which is chosen at random.

2. If there are intruding faces, the algorithm tries to work around them. The next face has to lie in the "double cone" generated by rotating the previous face around its leading edge (see Figure 3). Each face that intersects this double cone has its three vertices projected into the plane of the green circle in Figure 3, to determine what angle ranges around that circle must be thrown out. These ranges are aggregated, and an angle

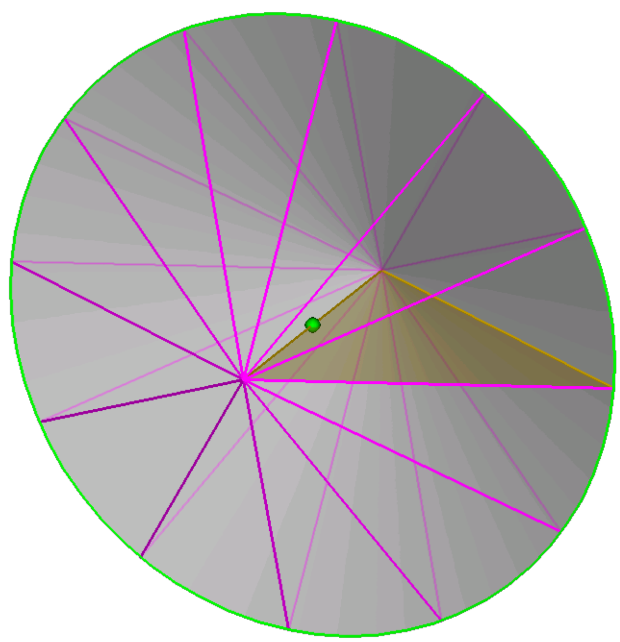

Figure 3. The "double cone" of possible faces (in purple) to add to the previous face (in orange).

of rotation is then randomly chosen with normal distribution in the complement of the aggregate.

After a new face is added, it is checked for intersection with the other faces in the current star and with any intruding faces. If an intersection is detected, the algorithm reverts the faces just added to the current star and tries again. After 25 failed attempts, it will add the current vertex to the non-completable stack and move to the next boundary vertex in the queue.

In practice, I have disabled the second method for adding faces because I found it to be too expensive. The program generates models with higher face counts using only the brute force of the first method.

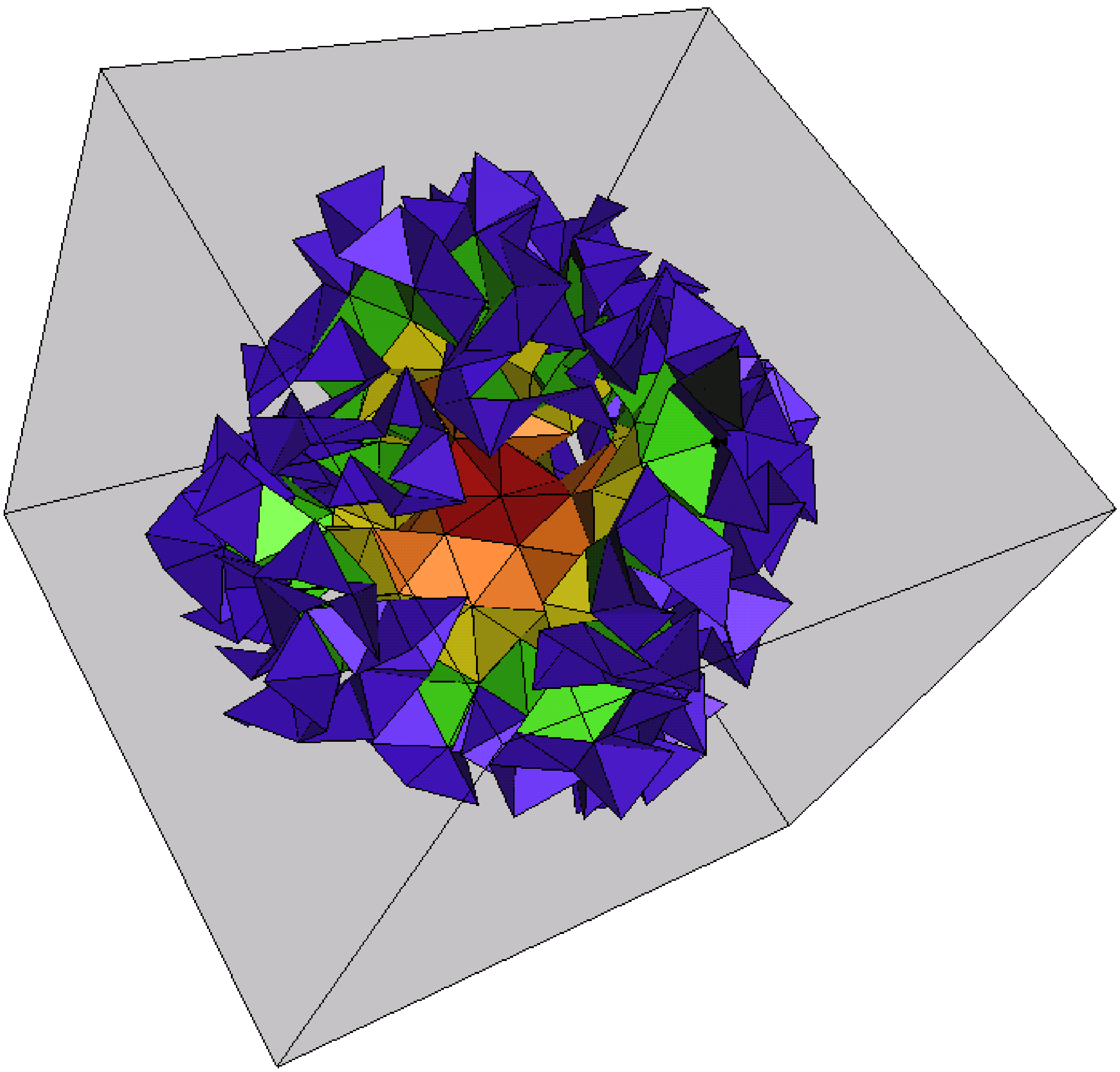

Figure 4. An incomplete model with 810 faces. Ring 3 (green) is complete, and Ring 4 (blue) is 73.3% complete.

Results

149 models with completed 3-rings and an incomplete ring with 810 faces (see Figure 4) were produced after running the program for 20 minutes in seed mode and 8 hours in production mode on an AMD Sempron 64 3000 (1.6 Ghz single core, 256 KB cache) with 2 GB memory. Unfortunately, I discovered after the run that my implementation of the rating function was incorrect, but had the same monotonicaty as the correct function, which explains why the program still worked.

Future Work

The second method for adding faces could be optimized so that it is not so expensive. One alternative to the method I presented is to have a set of premade templates that can be inserted if the intruding faces match some appropiate criteria (personal communications, Bryan Klingner).

Rerunning the program, especially with a longer seed phase, may generate a completed 4-ring, since the current best is only 86 faces away. A longer term project would be to prepare the program to run on a distributed cluster. Such an increase in processing power may be enough to find a model with a completed 5-ring.

References

1. Blanuša, D. Über die Einbettung hyperbolischer Räume in euklidische Räume. *Monatsh. Math. 59*, 1955, 217-229.
2. OGRE 3D: Open source graphics engine. http://www.ogre3d.org/