The curvature and dimension of a closed surface

S. Halayka*

October 8, 2019

Abstract

The curvature of a closed surface can lead to fractional dimension. In this paper, the properties of the 2-sphere surface of a three-dimensional ball and the 2.x-dimensional surface of a three-dimensional fractal set are considered. Tessellation is used to approximate each surface, primarily because the 2.x-dimensional surface of a three-dimensional fractal set is otherwise non-differentiable (having no well-defined surface normals).

1 Overview

Unlike in traditional geometry where dimension is an integer, fractional (non-integer) dimension occurs in fractal geometry. In fractal geometry, there are currently many ways to calculate the dimension of a surface [1, 2]. This paper uses a new method of calculating the fractional dimension of a surface – it is curvature that leads to this fractional dimension.

In this paper we will focus on the tessellation of closed surfaces. For instance, Marching Cubes [3, 4] can be used to generate triangular tessellations (meshes), where dimension $D \in (2.0, 3.0)$.

We will focus on the difference between the curvature and dimension of a 2-sphere and the 2.x-dimensional surface of a three-dimensional fractal set. We will generate both a 2-sphere and the 2.x-dimensional surface of a three-dimensional fractal set by using iterative quaternion equations. For example, a 2-sphere is generated by the iterative quaternion Julia set equation

$$Z = Z^2 + C,$$

but where $C = 0.0, 0.0, 0.0, 0.0$. Also for example, the 2.x-dimensional surface of a three-dimensional fractal set is generated by the iterative quaternion equation

$$Z = Z \cos(Z).$$

See [5] for information on how to perform quaternion multiplication, addition, cos, etc.

In the end, some notes are given.

*Sjhalayka@gmail.com*
2 The tessellation of a closed surface

Approximating the surface of a three-dimensional shape as a mesh allows us to calculate the surface’s dimension \( D \in (2.0, 3.0) \). This includes approximation of both a 2-sphere and the 2.\text{r}-dimensional surface of a three-dimensional fractal set.

First we calculate, for each triangle, the average dot product of the triangle’s face normal \( \hat{n}_i \) and its three neighbouring triangles’ face normals \( \hat{o}_1, \hat{o}_2, \hat{o}_3 \):

\[
d_i = \frac{\hat{n}_i \cdot \hat{o}_1 + \hat{n}_i \cdot \hat{o}_2 + \hat{n}_i \cdot \hat{o}_3}{3} \in (-1.0, 1.0].
\] (3)

Because we assume that there are three neighbours per triangle, the mesh must be closed (no cracks or holes, precisely two triangles per edge). The reason why the value \(-1.0\) is not achievable is because that would lead to intersecting triangles.

Then we calculate the normalized measure of curvature:

\[
k_i = \frac{1 - d_i}{2} \in [0.0, 1.0).
\] (4)

Once \( k_i \) has been calculated for all triangles, we can then calculate the average normalized measure of curvature \( K \), where \( t \) is the number of triangles in the mesh:

\[
K = \frac{1}{t} \sum_{i=1}^{t} k_i = \frac{k_1 + k_2 + \ldots + k_t}{t} \in (0.0, 1.0).
\] (5)

The reason why the value 0.0 is not achievable is because we are dealing with a closed surface, and so there’s bound to be some curvature.

The dimension of the closed surface is:

\[
D = 2 + K \in (2.0, 3.0).
\] (6)

As far as we know, this method of calculating the dimension of a closed surface is new [6, 7]. The entire C++ code for generating a mesh can be found at [8]. The entire C++ code for calculating a mesh’s dimension can be found at [9].

3 Vanishing versus non-vanishing curvature

Where \( r \in [2, \infty) \) is the integer sampling resolution, \( g_{\text{max}} \in (-\infty, \infty) \) is the sampling grid maximum extent, \( g_{\text{min}} \in (-\infty, \infty) \) is the sampling grid minimum extent, and \( g_{\text{max}} > g_{\text{min}} \), the Marching Cubes step size is:

\[
\ell = \frac{g_{\text{max}} - g_{\text{min}}}{r - 1} \in (0.0, \infty).
\] (7)

In this paper \( g_{\text{max}} = 1.5, g_{\text{min}} = -1.5 \), and \( r \) is variable.

For a 2-sphere, the local curvature all but vanishes as \( \ell \) decreases (as \( r \) increases):

\[
\lim_{\ell \to 0.0} K(\ell) = 0.0.
\] (8)
This results in a dimension of practically (but never quite) 2.0, which is to be expected from a non-fractal surface. See Figures 1 - 3.

On the other hand, for the 2.0-dimensional surface of a three-dimensional fractal set, the local curvature does not vanish as $\ell$ decreases:

$$\lim_{\ell \to 0} K(\ell) \neq 0.0.$$  

(9)

This results in a dimension considerably greater than 2.0, but not equal to or greater than 3.0, which is to be expected from a fractal surface. See Figures 4 - 7.

4 Notes

The minimum Marching Cubes step size, in real life, is the Planck length $\ell_P$.

Marching Squares [10, 11, 12] can be used to generate closed line paths, where dimension $D \in (1.0, 2.0)$. See Figures 8 - 10 for some examples of a line path. These figures might be helpful if there is difficulty envisioning the curvature in the case of Marching Cubes.

References


[8] https://github.com/sjhalayka/marching_cubes


[12] https://github.com/sjhalayka/Marching-Squares
Figure 1: Low resolution \((r = 10)\) surface for the iterative quaternion equation is \(Z = Z^2\). The surface’s dimension is 2.02.

Figure 2: Medium resolution \((r = 100)\) surface for the iterative quaternion equation is \(Z = Z^2\). The surface’s dimension is 2.06.

Figure 3: High resolution \((r = 1000)\) surface for the iterative quaternion equation is \(Z = Z^2\). The surface’s dimension is practically 2.0.
Figure 4: Low resolution ($r = 10$) surface for the iterative quaternion equation is $Z = Z \cos(Z)$. The surface’s dimension is 2.05.

Figure 5: Medium resolution ($r = 100$) surface for the iterative quaternion equation is $Z = Z \cos(Z)$. The surface’s dimension is 2.11.

Figure 6: High resolution ($r = 1000$) surface for the iterative quaternion equation is $Z = Z \cos(Z)$. The surface’s dimension is 2.08.
Figure 7: A two-dimensional slice of the iterative quaternion equation $Z = Z \cos(Z)$, showing the fractal nature of the set.
Figure 8: Example input (a two-dimensional greyscale image, consisting of pixels) and output (a 1.\(x\)-dimensional closed set of line segments) of the Marching Squares algorithm, approximating a 1-sphere (a circle), where sampling resolution is \(r = 8\). Note that for Marching Cubes, the input is a three-dimensional ‘greyscale image’, consisting of voxels, and the output is a 2.\(x\)-dimensional closed set of triangles.

Figure 9: Illustrated is a section of a closed line path, with surface normals. The average dot product of neighbouring line segments is \(d_i = 0.0\). This leads to a normalized measure of curvature \(k_i = (1 - d_i)/2 = 0.5\), which in turn leads to an average normalized measure of curvature \(K = 0.5\). The dimension is \(D = 1 + K = 1.5\).

Figure 10: A section of a closed line path as it goes from dimension 1.0 (at top) to 1.9999 (at bottom). In the end, where the dimension is 1.9999, the result is practically a rectangle. The reason why the dimension cannot be 2.0 is because that would lead to intersecting line segments.