The curvature and dimension of non-differentiable surfaces

S. Halayka*

January 7, 2019

Abstract

The curvature of a surface can lead to fractional dimension. In this paper, the properties of the 2-sphere surface of a 3D ball and the 2.x-surface of a 3D fractal set are considered. Tessellation is used to approximate each surface, primarily because the 2.x-surface of a 3D fractal set is otherwise non-differentiable.

1 Tessellation of surfaces

Approximating the surface of a 3D shape via triangular tessellation allows us to calculate the surface's dimension, somewhere between 2.0 and 3.0. In this paper, Marching Cubes [1] is used to generate the triangular tessellations.

For a 2-sphere, the *local* curvature vanishes as the size of the triangles decreases. This results in a dimension of 2.0. See Figures 1, 2, and 3.

On the other hand, for the surface of a 3D fractal set, the local curvature does not vanish. This results in a dimension greater than 2.0, but no greater than 3.0. See Figures 4, 5, 6, and 7.

A small piece of C++ code is given in the next section, in lieu of mathematical notation.

References

[1] Bourke P. `http://paulbourke.net/geometry/polygonise/`

*sjhalayka@gmail.com

2 Core C++ code

```
int main(int argc, char **argv)
{
        if (2 != argc)
                return 1;

        indexed_mesh mesh;

        if (false == mesh.load_from_binary_stereo_lithography_file(argv[1]))
                return 2;

        vector< vector<size_t> > tri_neighbours;
        vector<vertex_3> tri_normals;

        tri_neighbours.resize(mesh.triangles.size());
        tri_normals.resize(mesh.triangles.size());

        for (size_t i = 0; i < mesh.triangles.size(); i++)
        {
                mesh.get_tri_neighbours(i, tri_neighbours[i]);
                tri_normals[i] = mesh.get_tri_normal(i);
        }

        float final_measure = 0;

        for (size_t i = 0; i < mesh.triangles.size(); i++)
        {
                vertex_3 n0 = tri_normals[i];
                vertex_3 n1 = tri_normals[tri_neighbours[i][0]];
                vertex_3 n2 = tri_normals[tri_neighbours[i][1]];
                vertex_3 n3 = tri_normals[tri_neighbours[i][2]];

                float dot1 = n0.dot(n1);
                float dot2 = n0.dot(n2);
                float dot3 = n0.dot(n3);

                float d = (dot1 + dot2 + dot3) / 3.0f;
                float measure = (1.0f - d) / 2.0f;

                final_measure += measure;
        }

        cout << "Dim:" << 2.0f + final_measure/mesh.triangles.size() << endl;

        return 0;
}
```

Figure 1: Low resolution surface for the iterative equation is $Z = Z^2$. The surface's dimension is 2.01682.

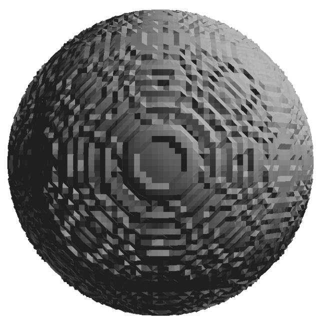

Figure 2: Medium resolution surface for the iterative equation is $Z = Z^2$. The surface's dimension is 2.05516.

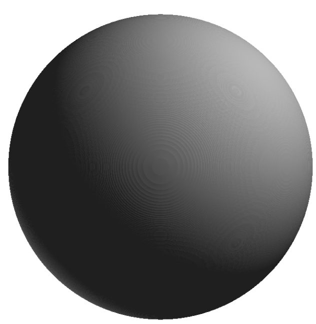

Figure 3: High resolution surface for the iterative equation is $Z = Z^2$. The surface's dimension is 2.00097.

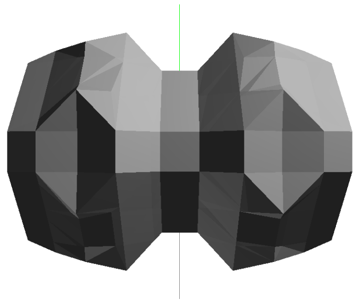

Figure 4: Low resolution surface for the iterative equation is $Z = Z\cos(Z)$. The surface's dimension is 2.05266.

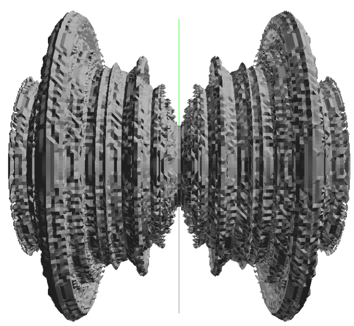

Figure 5: Medium resolution surface for the iterative equation is $Z = Z\cos(Z)$. The surface's dimension is 2.10773.

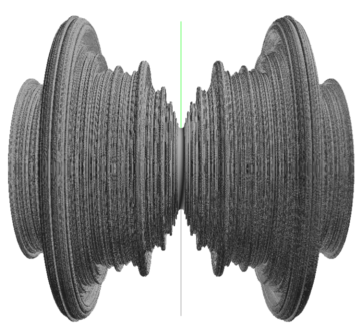

Figure 6: High resolution surface for the iterative equation is $Z = Z\cos(Z)$. The surface's dimension is 2.07679.

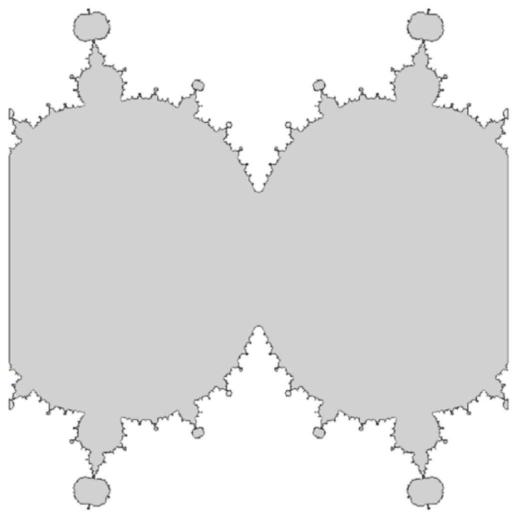

Figure 7: A 2D slice of $Z = Z\cos(Z)$, showing the fractal nature of the set.