ADSODA Conversion and Extension

Fabian Junge

October 26, 2012

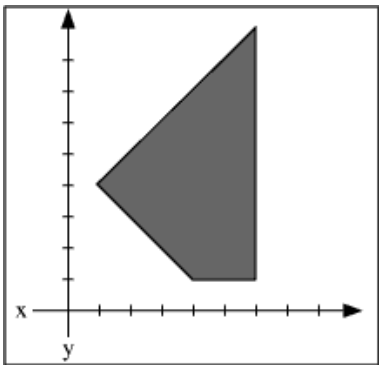
1 Introduction

When the class began, I was most interested in 4 dimensional visualization. My first project idea was to have a system of hypercubes casting shadows on a floor. As the class went on, and the Sierpinski Triangle, Pyramid, and Carpet came up these also fascinated me, and the first program I actually got to run in the cube was a (malformed) Sierpinski Pyramid. So, this project will attempt to marry the two, and create a four dimensional representation of the Cantor set as, analogously the Sierpinski Carpet is a two dimensional representation and the Menger Sponge is a three dimensional representation. To accomplish this, I will revive Greg Ferrar's old project, ADSODA, to work with modern C++ and the modern cube.

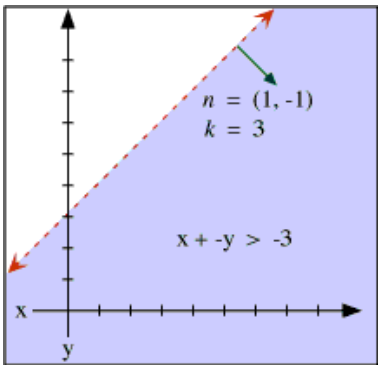
2 ADSODA

2.1 So, what is ADSODA?

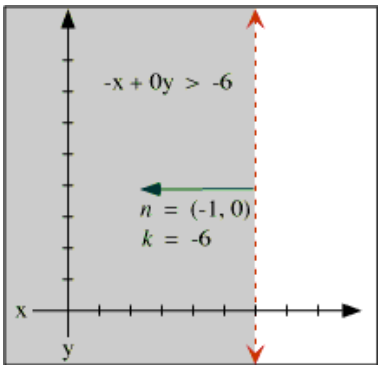
ADSODA stands for Arbitrary-Dimensional Solid Object Display Algorithm and is a system for rendering and displaying solids of arbitrary dimension to a space of arbitrary dimension. In our case, we are only interested in the two dimensional space (our computer screens) and the three dimensional space (the Cube). How ADSODA achieves this is by providing a different way of representing solids, specifically as an intersection of half-spaces. Half-spaces in ADSODA are represented by their equation, normal vector, and a constant. The following images, from the ADSODA website [1], demonstrate this concept:



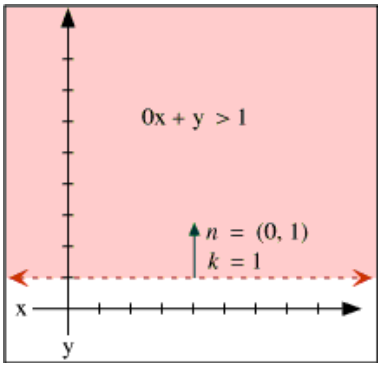
(a)



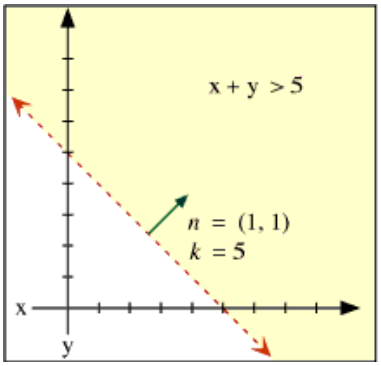
(b)



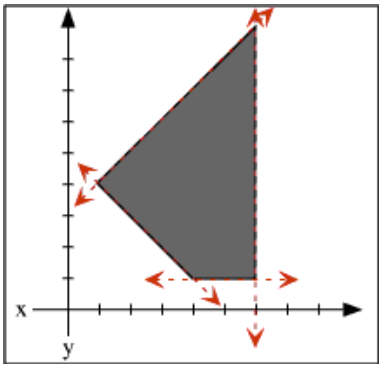
(c)



(d)



(e)



(f)

Figure 1: ADSODA Half-spaces Representation[1]

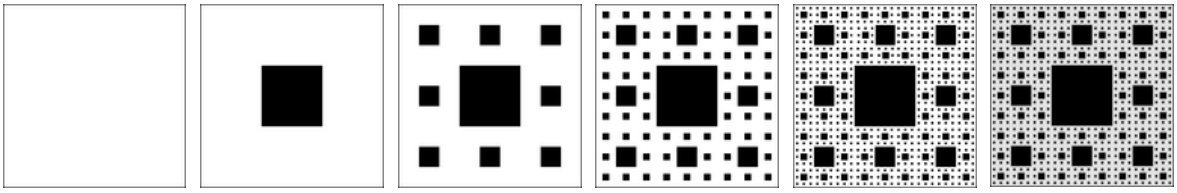
2.2 Why is this useful?

In addition to this representation of a solid, ADSODA provides methods for projecting, rendering, and lighting such solids in two or three space. While such methods definitely exist for our usual representation of solids (as a collection of points) for intersections-of-half-spaces these methods are simpler and easier to calculate, at the cost of being unable to define concave shapes. This is something I am hoping to improve upon by modifying ADSODA.

3 The Cantor Set

The Cantor set is the sire of interesting geometrical figures as the Sierpinski Triangle, Pyramid, Carpet and the Menger Sponge. The Cantor set has many fascinating qualities, but a surprisingly simple construction. Take the interval $[0, 1]$ on the real number line, and remove the middle third. Then remove the middle thirds from the remaining line segments, ad infinitum. It should be trivial note how this algorithm of removing the middle third translates the Sierpinski Carpet and from there to the Menger sponge. Below are visuals from Wikipedia to illustrate this point:

Figure 2: A visual representation of the first 6 iterations of the Cantor Set[2]



(a) Step 0 (b) Step 1 (c) Step 2 (d) Step 3 (e) Step 4 (f) Step 5

Figure 3: The first five steps of the Sierpinski Carpet[3]

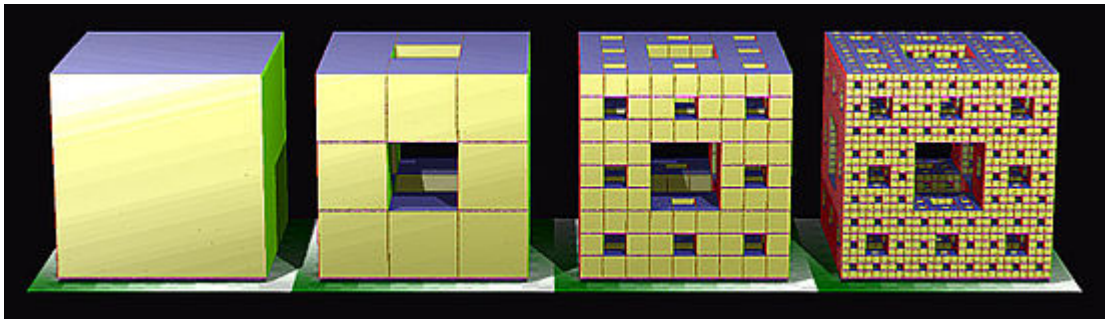


Figure 4: The first three steps of the Menger Sponge[4]

4 My Goal

So, given this background information of the Cantor Set and ADSODA, my goal is threefold. First, I would like to revive ADSODA to work with C++ and the modern Cube. Second, I would like to create these visual representations of the Cantor set in the Cube, using ADSODA. Third, and ultimately, I would like to construct a four dimensional representation of the Cantor Set, along the lines of the Sierpinski Carpet and the Menger Sponge, by removing from a hypercube smaller hypercubes on each of it's 8 sides and the center and then bring that to the Cube using ADSODA.

References

[1] Greg Ferrar
ADSODA: Arbitrary-Dimensional Solid Object Display Algorithm
`http://www.flowerfire.com/ADSODA/`

[2] Wikipedia
The Cantor Set
`http://en.wikipedia.org/wiki/Cantor_set`

[3] Wikipedia
The Sierpinski Carpet
`http://en.wikipedia.org/wiki/Sierpinski_carpet`

[4] Wikipedia
The Menger Sponge
`http://en.wikipedia.org/wiki/Menger_sponge`