Cantor Set Analysis and Visualization

In 2, 3, and 4 dimensions, using ADSODA

Fabian Junge

Friday November 9th, 2012

The Cantor Set

The Cantor Set is a set of points on a line segment, and a very interesting fractal. It is constructed in the following manner:

1. Consider a line segment.
2. Remove the middle third of this line segment.
3. Consider each remaining line segment, and apply this process.
4. Continue to do so infinitely.

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

Two Dimensions

For the two dimensional versions of the Cantor Set, simply imagine a Cantor Set along the x-axis, and a Cantor Set along the y-axis as below. There are now two ways to construct the 2d analog.

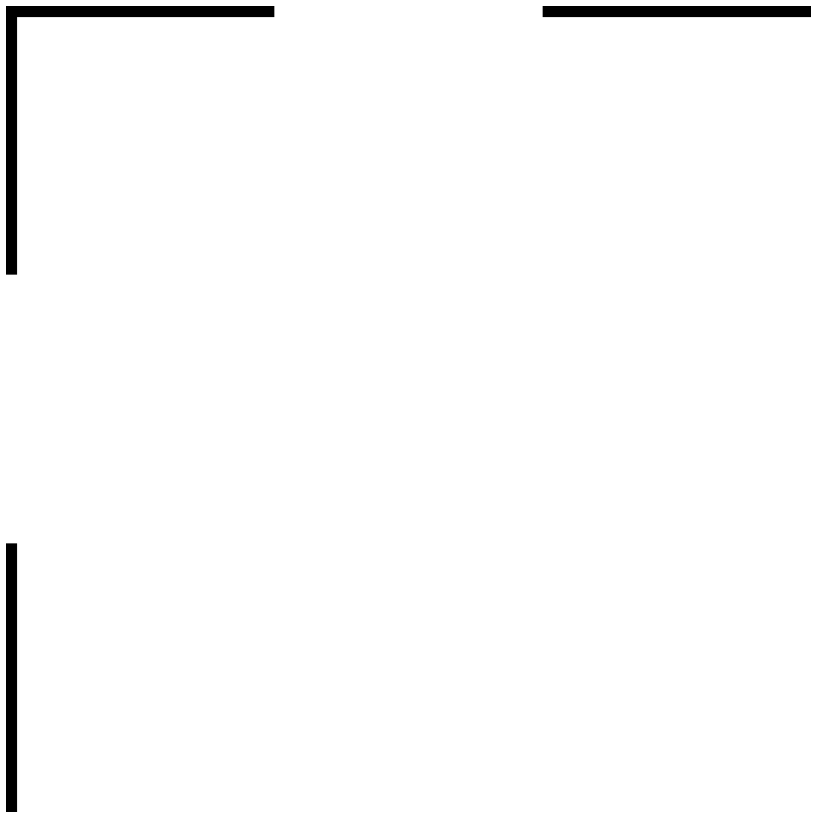
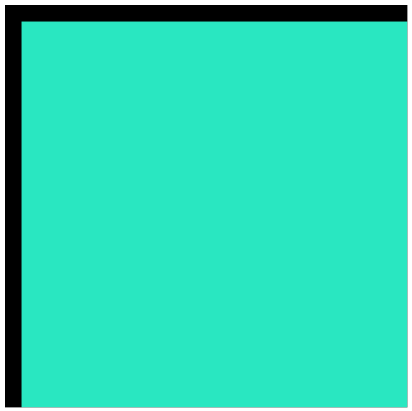
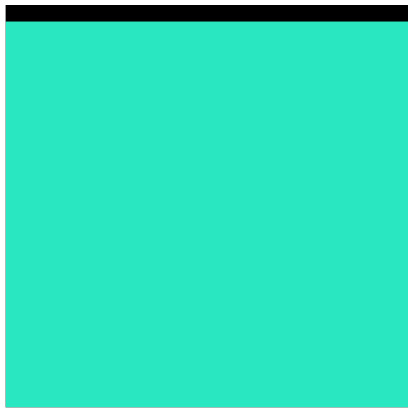
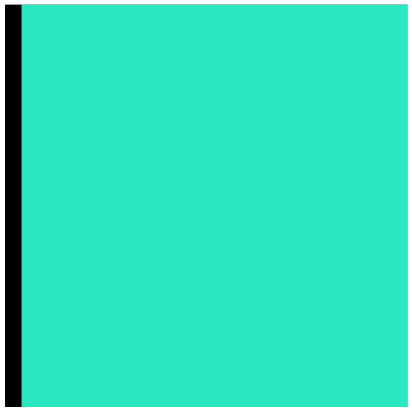


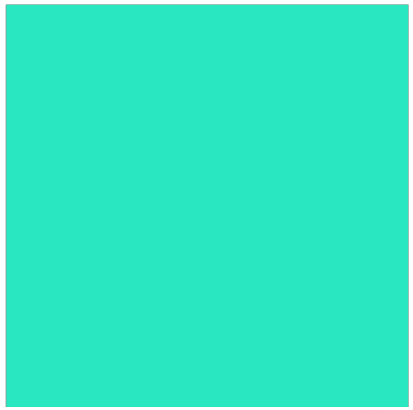
Figure: 2-d Cantor Setup

The Intersection

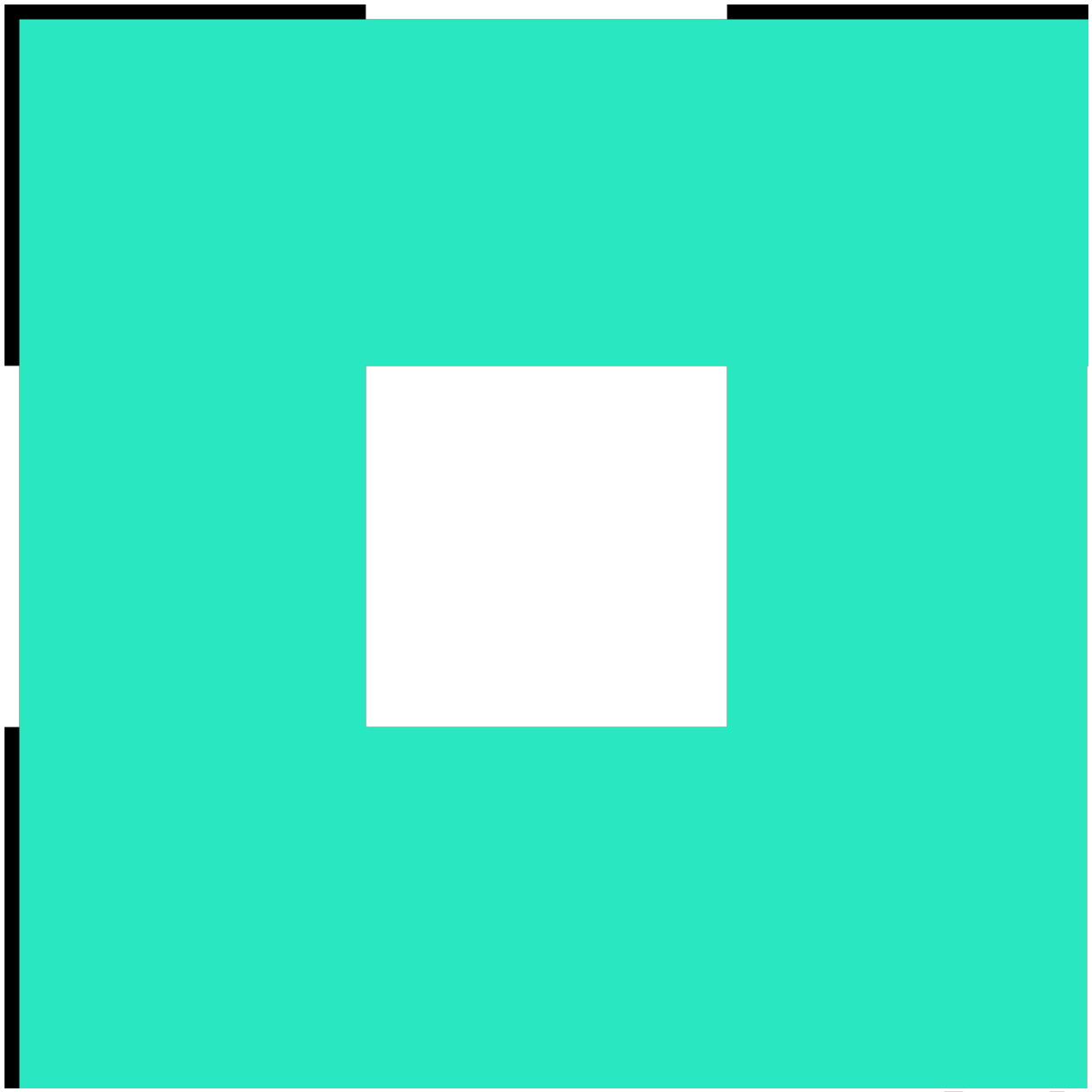








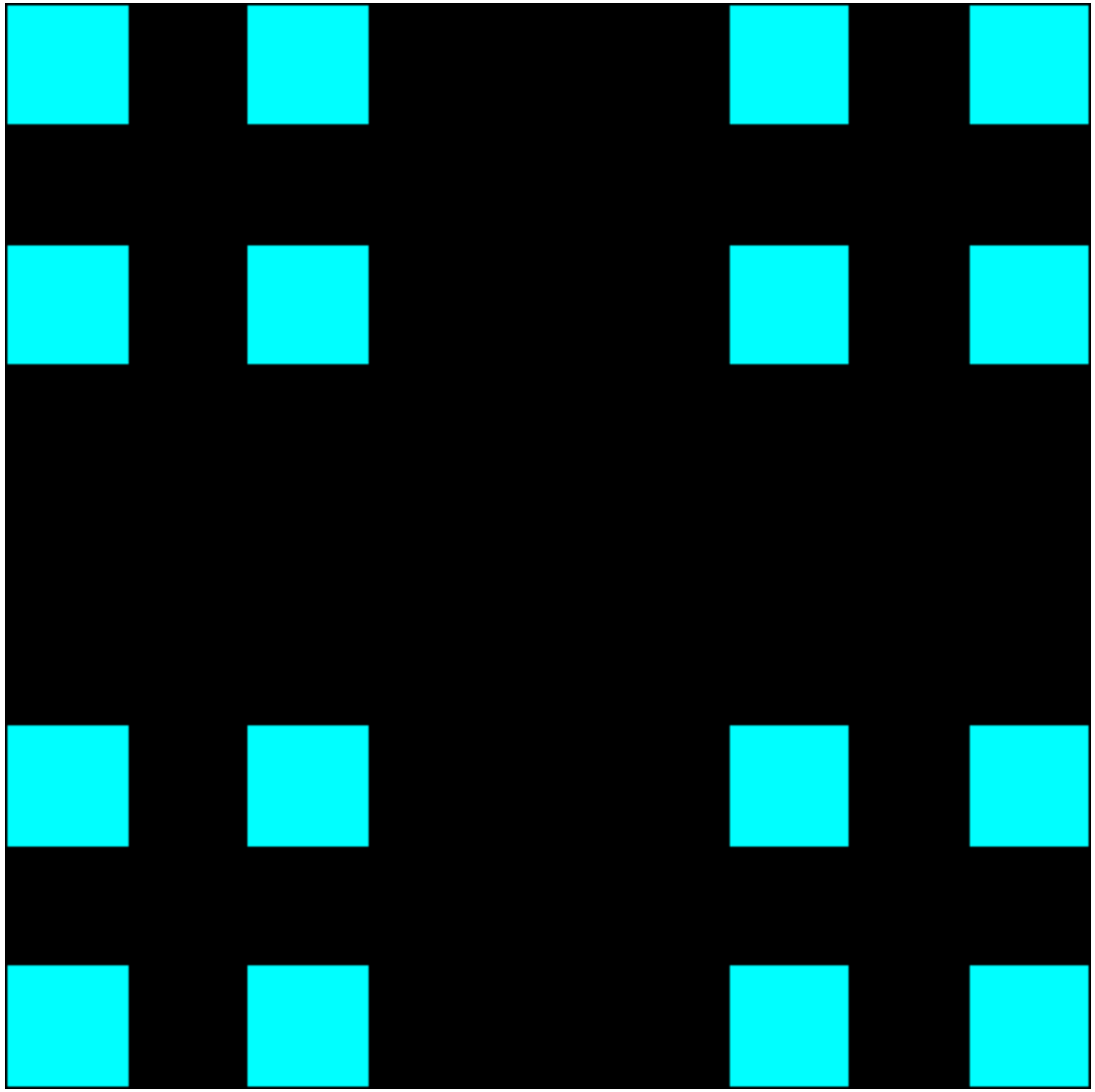
The Union



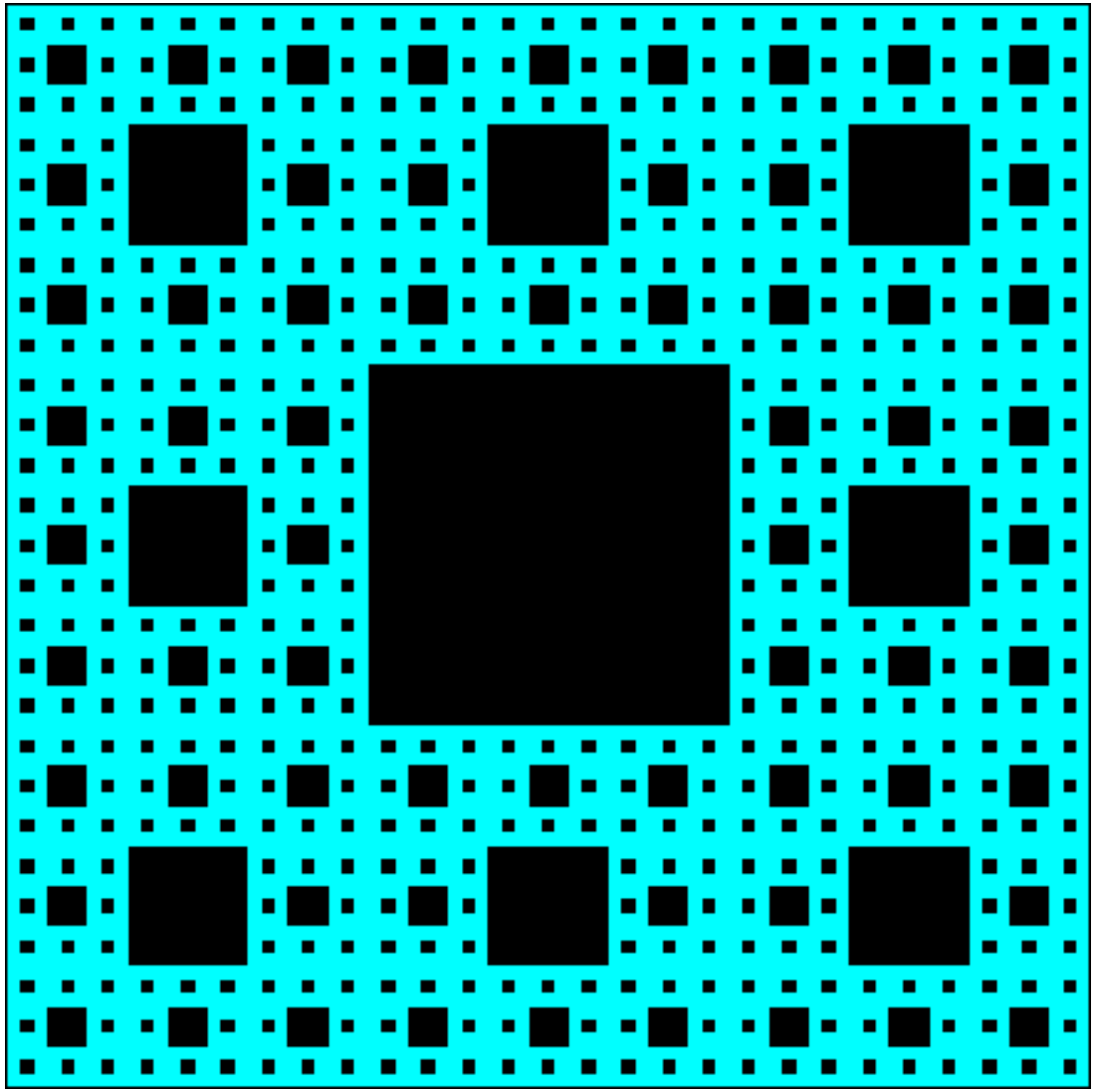
Fractal Results

So, the two dimensional Cantor Sets are created by taking two Cantor Sets, one for each axis, and taking either their union or their intersection. To continue building the fractals, simply apply the Union / Intersection procedure to each of the smaller squares created. (There are 4 squares of 1/3 the side length created by the intersection of Cantor Sets, and 8 by the Union of Cantor Sets). These figures are also established fractals in their own right.

Two Dimensional Cantor Dust



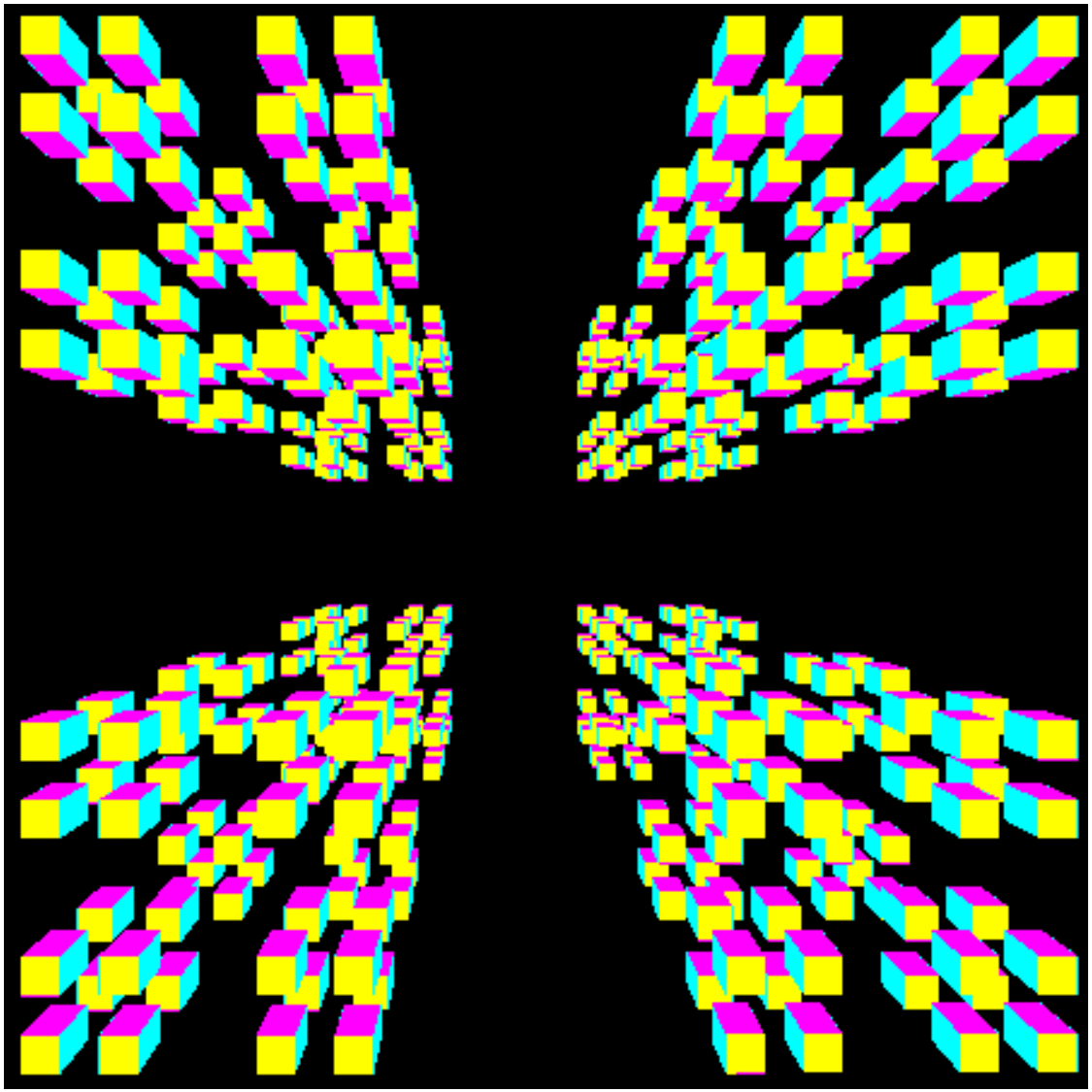
Sierpinski Carpet



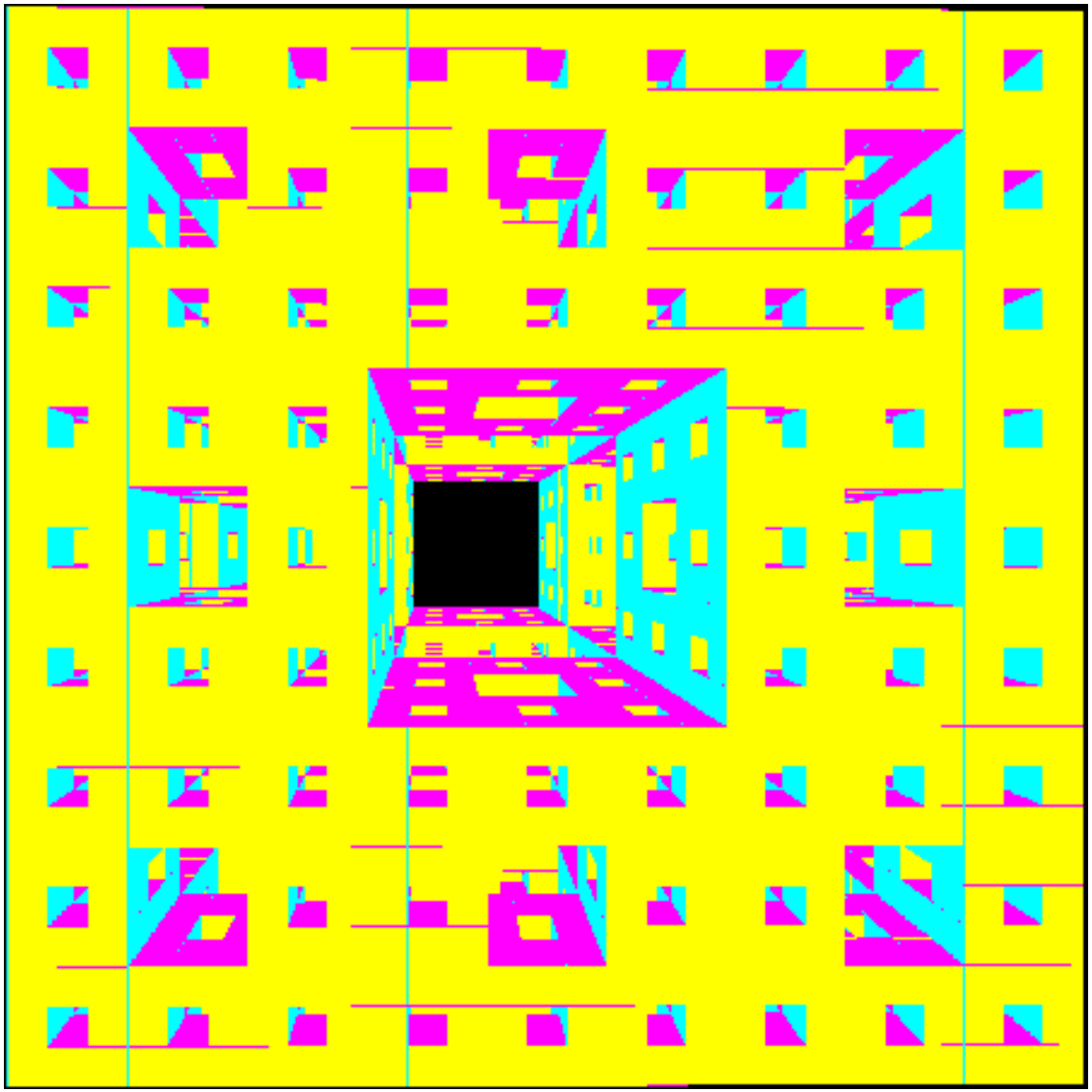
Three Dimensions

For three dimensions, we can simply expand on the idea used in two dimensions: We will have a Cantor Set on the X, Y, and Z axies. Now, We can take the intersection of all of these Cantor Sets and get what is known as Three Dimensional Cantor Dust (pictured later). We can take the intersection and get something that is rather boring to look at, as it is a complete cube with the center cut out, and the centers of the remaining cube cut out, and so forth (not pictured later). We can also do something more interesting in three dimensions, we can take the Union of the Intersection (or the Intersection of the Union, mathematically it is the same). This results in the very interesting fractal, the Menger Sponge;

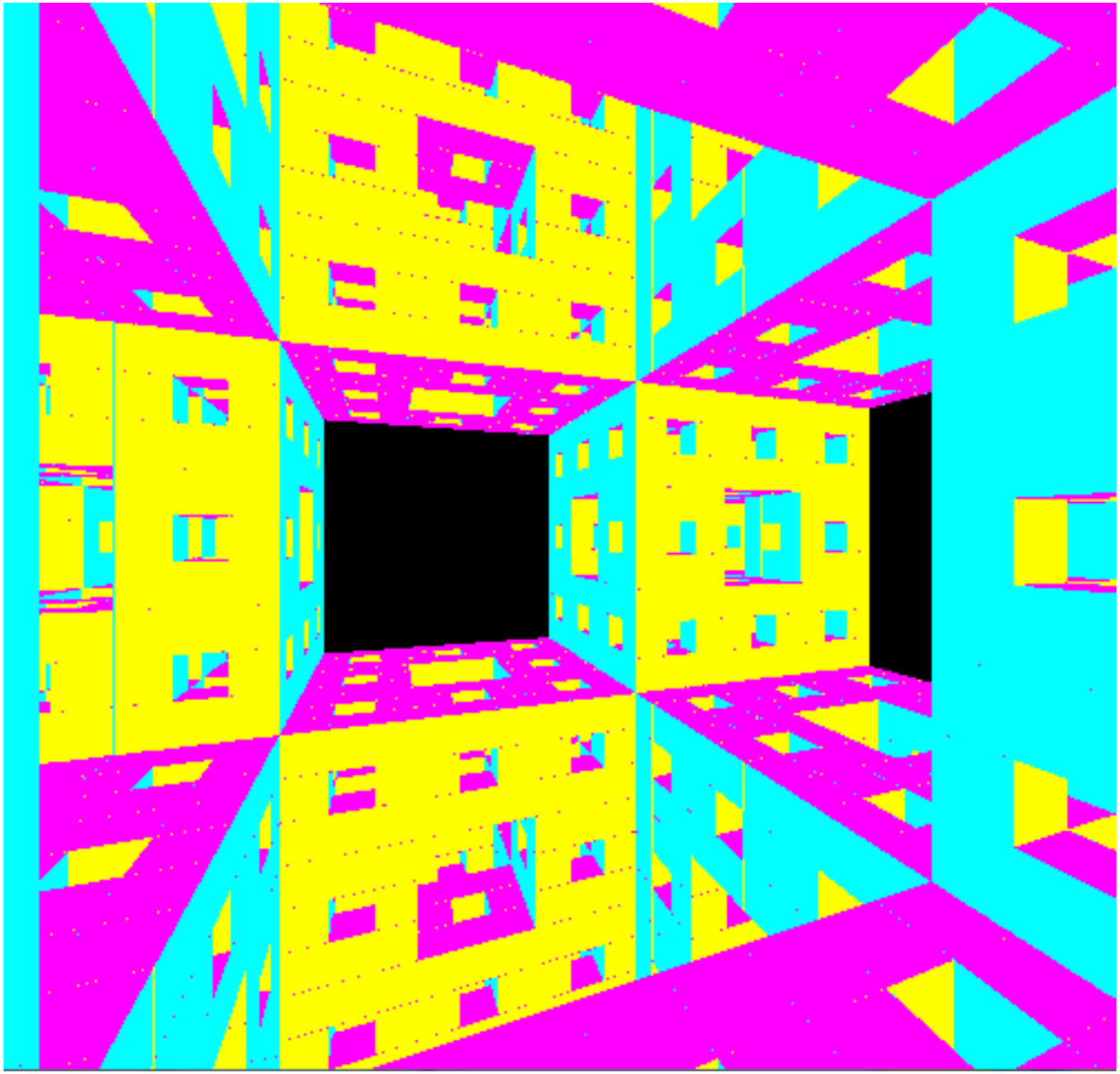
Three Dimensional Cantor Dust



Menger Sponge



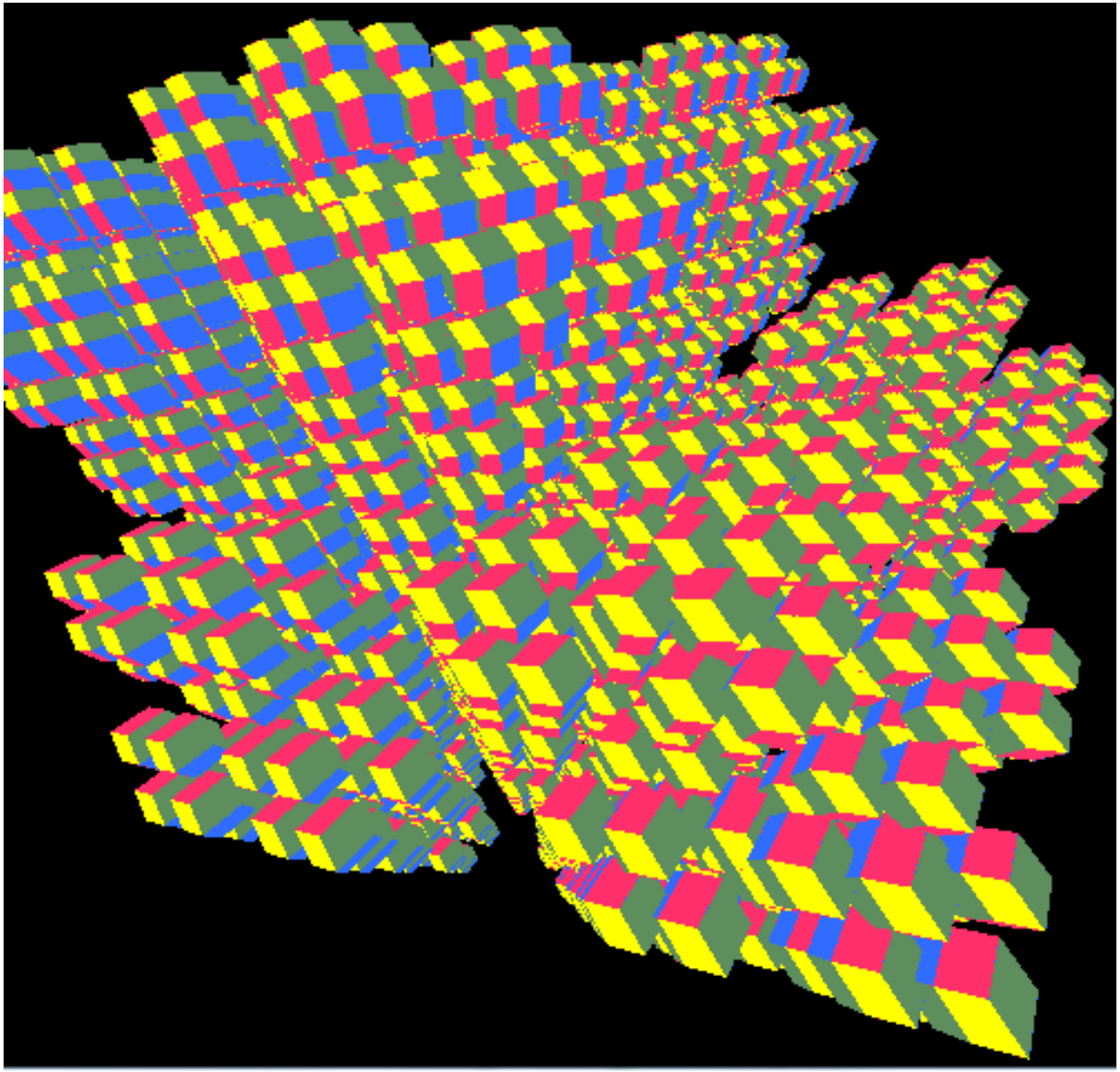
Menger Sponge



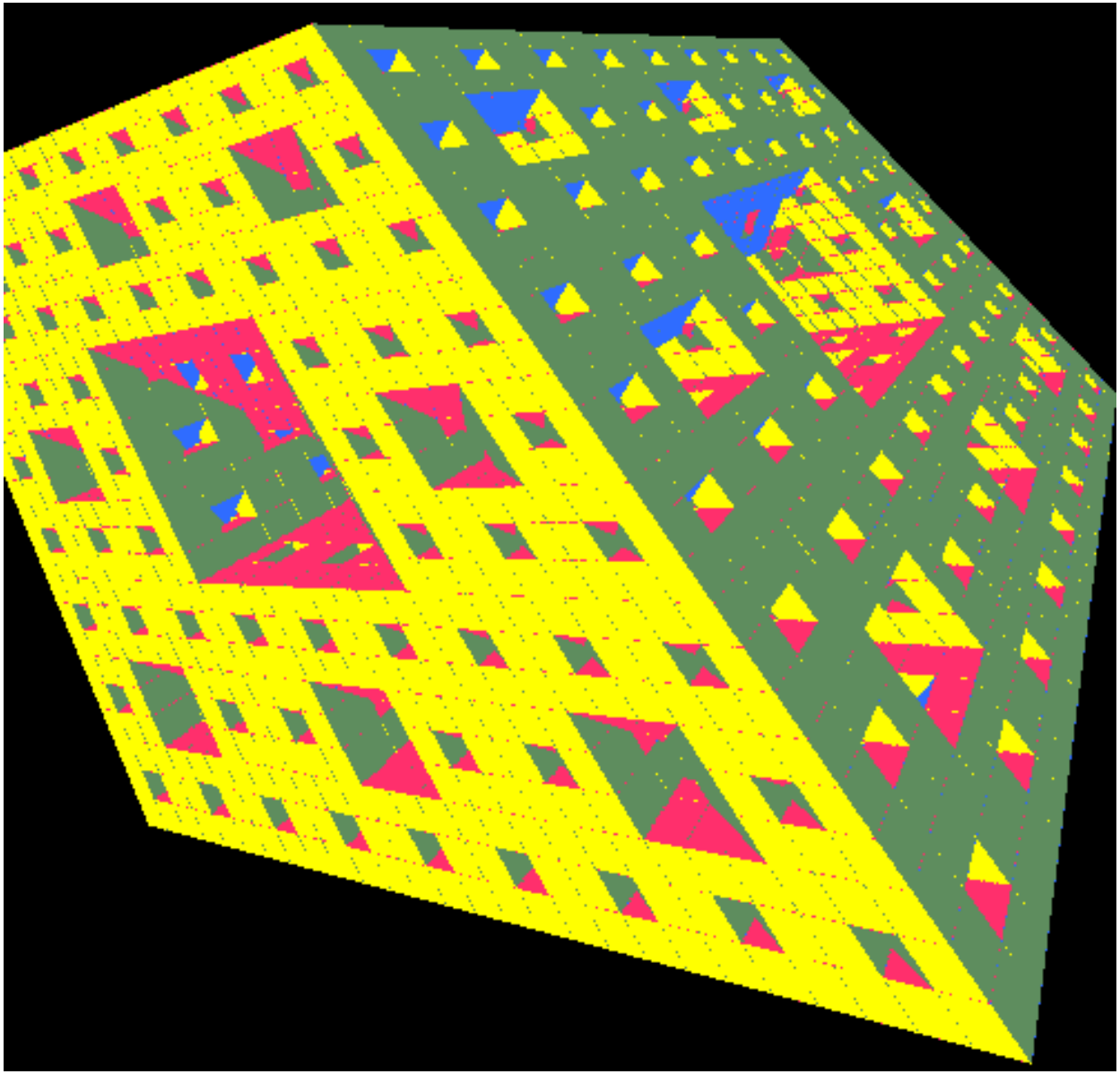
Four Dimensions

What we did in three dimensions we can easily extend to four and we end up with four analogs.
To actually be able to visualize these, we project them into three dimensions. Curiously enough, they end up looking exactly like the three dimensional versions, as our default orientation of the 4-cubes used look like 3-cubes when projected. Therefore, we rotate the hypercubes a bit so that they look interesting in three dimensions. Two of the resulting analogs are removing things inside the main hypercube, and therefore are not pictured as there is not much to see. Why and how we get these four and why there are four will be explained right after these pictures.

Four Dimensional Cantor Dust



Four Dimensional Cantor Sponge



Quick Definition

Consider a line segment. You can cut this segment into three parts of equal length. Label the parts 0, 1, and 2 respectively. The first iteration of the Cantor Set will consist of the parts labelled 0 and 2. You can, in fact, classify each line segment you process while creating the Cantor Set this way.
That is, for the 1 dimensional Cantor Set, you can have the following boolean equation:
$c := \mathbf{Pos} == 0 \,||\, \mathbf{Pos} == 2 \leftrightarrow \mathbf{Pos} \,! = 1$
We can use this definition to construct our higher dimensional analogs.

Two Dimensions

For the two dimensional case, we have two axies, X and Y. Let the boolean x denote that this square's X position is in the Cantor Set on the X axis, and let the boolean variable y denote that its Y position is in the Cantor Set on the Y axis. That is, $x := \mathbf{XPos}! = 1$ and $y := \mathbf{YPos}! = 1$.
Some quick syntax to make this more compact, let $x + y$ denote "x OR y" and let xy denote "x AND y".
In two dimensions we have the options of $x + y$ (The Union, aka the Cantor Dust) or xy (The Intersection, aka the Sierpinski Carpet).

Two Dimensions Con.

We can construct the following table at each level of construction.

xPos	yPos	$x + y$	xy
0	0	T	T
0	1	T	F
0	2	T	T
1	0	T	F
1	1	F	F
1	2	T	F
2	0	T	T
2	1	T	F
2	2	T	T

From this we can gather that the Sierpinksi Carpet $(x + y)$ has a Hausdorff dimensions of $\frac{\log(8)}{\log(3)} = 1.8928$ and the Cantor Dust has a Hausdorff dimension of $\frac{\log(4)}{\log(3)} = 1.2619$

Three Dimensions

With three dimensions we end up with the three equations:
$x + y + z$,
xyz, and
$xy + zy + xz$ (which is equivalent to $(x + y)(z + y)(x + z)$ and is what creates the Menger Sponge).
In general, the equations we get are the combinations of the variables, x and y by ORs and ANDs, where the order in which teh ORs and ANDs is applied does not matter. For higher dimensions we have more variables, and therefore more combinations.

Four Dimensions

With four dimensions we end up with the four equations:
$x + y + z + w$,
$xyzw$,
$xyz + wyz + xwz + xyw$, and
$(x + y + z)(w + y + z)(x + w + z)(x + y + w)$.

What is ADSODA?

- An old project by Greg Ferrar, a former graduate student.
- Written in rather old code for a very different implementation cube.
- ADSODA Stands for "Arbitrary-Dimensional Solid Object Display Algorithm".
- It's purpose is to display solid objects of arbitrary dimension on a viewing space of any dimension. For us, this means it is a way of displaying 4+ dimensional objects on a 2d viewing screen (a monitor) or a 3d viewing environment (the cube).

What does ADSODA do?

ADSODA comes with a collection of built in goodies:

- n-dimensional Lighting.
- n-dimensional Hidden Solid Object Removal.
- n to $(n-1)$ dimensional Projection.
- 2D and 3D rendering.

ADSODA does not currently work, but part of my project is to get it working for C++03 and the UIUC cube.

How does ADSODA do this?

- ADSODA has a different internal representation for solids than openGL.
- A n-dimensional solid in ADSODA is defined as the intersection of several n-dimensional halfspaces.
- Each halfspace is represented by it's equation (from which the normal can be quickly, O(1), derived) and a constant.
- Next Slide: Pictures!

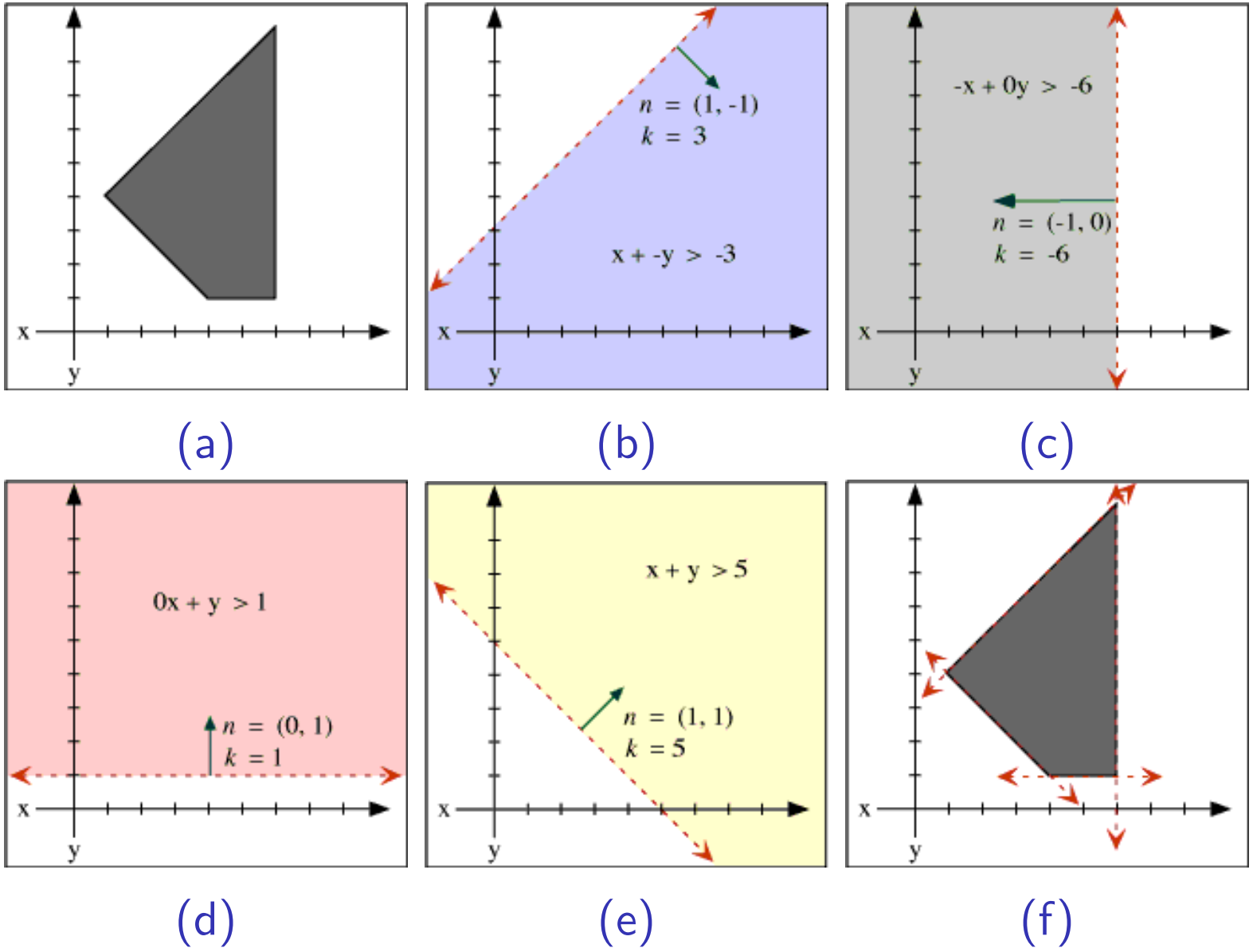


Figure: ADSODA Half-spaces Representation[1]

Sources

- Greg Ferrar
  ADSODA: Arbitrary-Dimensional Solid Object Display Algorithm
  http://www.flowerfire.com/ADSODA/
- Wikipedia
  The Cantor Set
  http://en.wikipedia.org/wiki/Cantor_set
- Wikipedia
  The Sierpinski Carpet
  http://en.wikipedia.org/wiki/Sierpinski_carpet
- Wikipedia
  The Menger Sponge
  http://en.wikipedia.org/wiki/Menger_sponge
- Wikipedia
  List of fractals by Hausdorff dimension
  http://en.wikipedia.org/wiki/List_of_fractals_by_Hausdorff_dimension