An Algorithm for Extending Menger-Type Fractal Structures

Manuel Diaz Regueiro[1] and Luis Diaz Allegue[2]

[1]Retired Math Teacher, Spain; mdregueiro2@gmail.com

[2]ANFACO-CECOPESCA, Department of Circular Economy, Spain; ldiaz@anfaco.es

Abstract

In this paper, the potential to generate a wide range of designs using an algorithm that creates Menger-type fractal constructions is explored. The authors describe a method that can be used to build a Menger-type figure from which, the initial "atoms", the rules and distances can be changed to create many interesting figures. Furthermore, this method can be applied in a new variation called Menger-Diaz. The Menger-Diaz variation is created by modifying the way the atoms are placed, not removed, at each level, resulting in a shape with unique properties and characteristics that are demonstrated through several examples. Therefore, this method can lead to applications of these designs in artistic works.

Introduction

In the world of ideas many wonders are awaiting us. Sierpinski- or Menger-like designs are a well-known and popular class of experimental designs that were first proposed by the mathematician Karl Menger in 1926 [1]. Menger Sponge (or cube) is a prime example of a fractal, a type of geometric shape that exhibits self-similarity at different scales [2].

The Menger Sponge has several applications in fields such as computer graphics, nanomaterials and data visualization and has been studied in fields such as mathematics, physics, and engineering, due to its unique properties and the insights it can provide into the behavior of complex systems [2]–[4].

In particular, the Menger Sponge is a three-dimensional fractal shape formed by a cube in which each face is divided into 9 initial "atoms" and subsequently iteratively removing the central "atom". This process can be repeated indefinitely, resulting in a shape with an infinite number of faces and an infinitely complex surface. The number of times the fractal creation process has been applied is normally called levels. As more levels are added, the fractal becomes more complex. However, as levels increase, the complexity of the fractal also increases, and it takes more time and resources to compute it.

In addition to the possibility of changing the first "atom" and the level of the Merger figures, we can also change their distances. In summary, two types of distance are relevant: fixed distances and variable distances. Fixed distances are established at the beginning of the construction process and do not change as the level increases. These distances are typically related to the size of the first "atom". The fixed distances in Menger figures are important because they determine the overall size and shape of the figure, as well as the relative positions of the small cubes within it. Variable distances, on the other hand, are distances that change as the recursion level increases. In the case of the Menger Sponge, these distances are related to the number of "atoms" that lie between any two points within the figure.

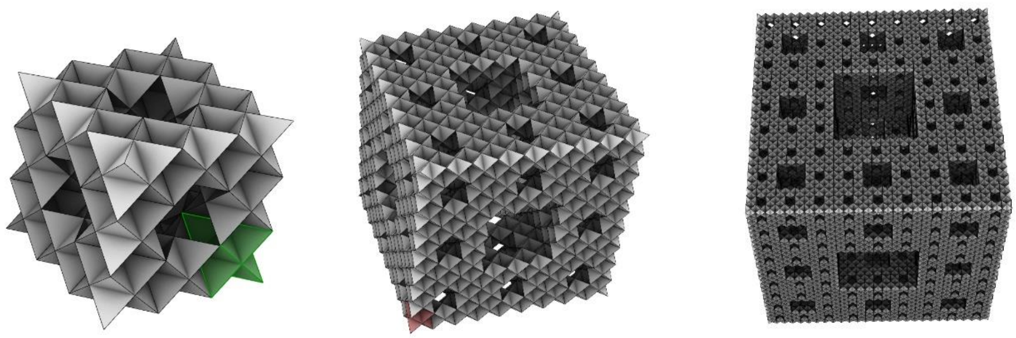
Overall, both fixed- and variable-distances play important roles in the structure and properties of Menger figures. Nonetheless, in this paper, when we discuss distance, we are referring to variable distance, which determines the complexity and interconnectedness of the initial "atom" within it.

In this paper, we explore the prospect of generating a plethora of amazing designs using an algorithm we discovered that extends, in many ways, the Menger fractal constructions.

In previous work, it was discovered that the Sierpinski tetrahedron with Stella Octangula has a unique property in that it can be used to fit both polyhedra made of tetrahedra, such as the Sierpinski tetrahedron, and polyhedra containing cubes, such as Menger Sponge. This is because the vertices of the Stella in a plane form a square, and the Stella occupies the volume of a cube. Thus, all figures with cubes can be substituted by Stella. Therefore, to make a Menger Sponge with a Stella Octangula, first we needed an algorithm to build it. The first idea was to encode in a vector list the existence or not of the cube in each of the 27 positions where it could be. As the program is written in Python, the nested vector list is written as follows:

a= [[[1,1,1], [1,0,1], [1,1,1]], [[1,0,1],[0,0,0],[1,0,1]],[[1,1,1],[1,0,1],[1,1,1]]] (*Menger*)

which reflects a Menger Sponge. In each of the cubic positions, we place the result of the previous iteration according to whether the corresponding number of the vector list is 1 or 0. The first three levels of a Menger Sponge with Stella Octangula as the initial "atom" can be seen in Figure 1.



(a) (b) (c)

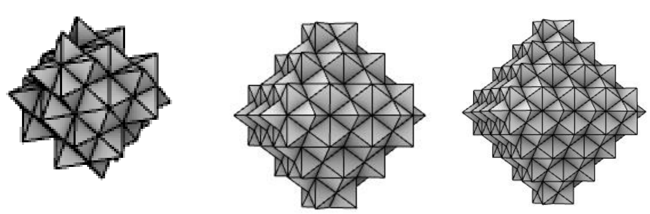
Figure 1: *From left to right, the first three levels of the Menger Sponge are represented with Stella Octangula as the initial "atom".*

The 'complement' Menger figure

The 'complement' Menger figure is obtained by replacing each of the initial "atoms" in the Menger Sponge with a hole and the hole with an "atom". By changing the 1's to 0's and the 0's to 1's in the Menger figure, we can create a "complement" Menger figure. This means that if the original Menger figure is a cube, the complement figure will be an octahedron (the dual of a cube). This holds whether the distance between the figures is maintained in each iteration and whether the figures are allowed to intersect. Therefore, many different matrices can be created. For example, the following one:

a=[[[0,0,0],[0,1,0],[0,0,0]],[[0,1,0],[1,1,1],[0,1,0]],[[0,0,0],[0,1,0],[0,0,0]]] (Menger 2)

To illustrate this, we apply this vector list and add an Escher's Solid polyhedron (Figure 2).

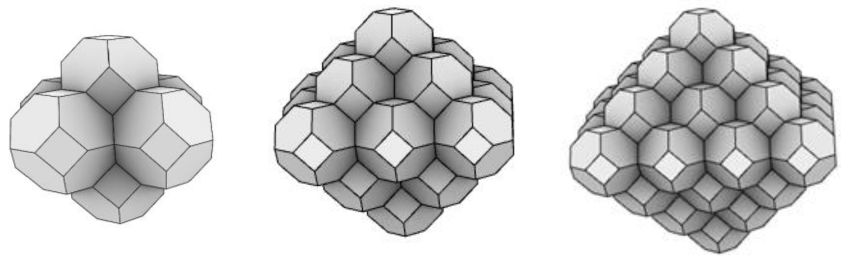


(a) (b) (c)

Figure 2: *Menger "complement" figure (with Escher's Solid polyhedron) at levels 1 (a), 2 (b), and 3 (c). Without changing distances in the process of recursion.*

If we change the vector list again and use a truncated octahedron as the initial "atom", the result is shown in Figure 3. Interesting only with distances that do not change in each level:

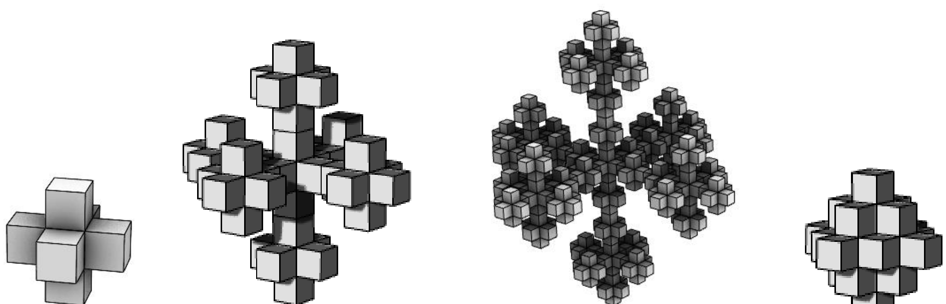
a= [[[0,1,0],[0,0,0],[0,1,0]],[[0,0,0],[1,0,1],[0,0,0]],[[0,1,0],[0,0,0],[0,1,0]]](*Menger 3*)



(a) (b) (c)

Figure 3: *Modified Menger figure (with truncated octahedrons) at levels 1 (a) 2 (b) and 3 (c).*

In Figure 4, we can see a "complement" Menger figure (made up of cube "atoms") that is either constructed with a distance on which figures are contiguously touching each other in a pure Menger style, or with a distance equal in each level, being able to intersect the figures in each new level. From now on, these two types of distance can be referred to as distance type 1 and distance type 2, respectively.



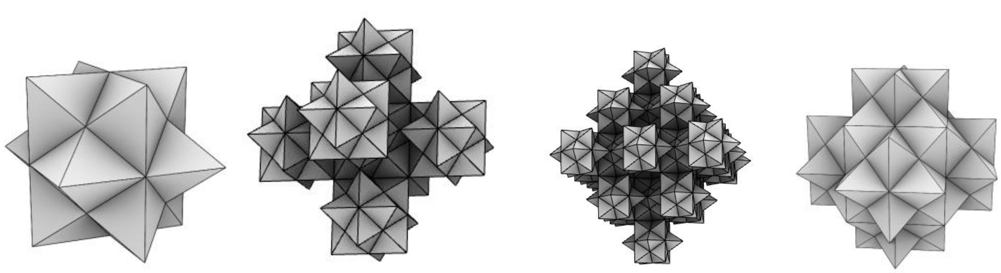
(a) (b) (c) (d)

Figure 4: *Menger "complement" figure with distance type 1 (a, b and c) and levels 1, 2 and 3. And type 2 (d) dist=cube's side.*

Web article [10] describes complement and figure 4 type 1 before us.

Playing games with distances

As we briefly mentioned above, one way to "play" with Menger-type figure configuration is to vary the distance between the "atoms" as the level increases. By doing this, it is possible to create a wide range of different Menger figures, each with its unique properties. For example, increasing the distance between the "atoms" at higher levels can result in a Menger figure with a more open, airy structure, while decreasing the distance between the small cubes can result in a Menger figure with a more compact, dense structure. These distance changes offer figures in a non-classical Merger style. In Figure 5 we show the different results using an Escher's solid polyhedron and different levels. Escher's solid polyhedron is the First stellation of the rhombic dodecahedron and appears in Escher's Waterfall.

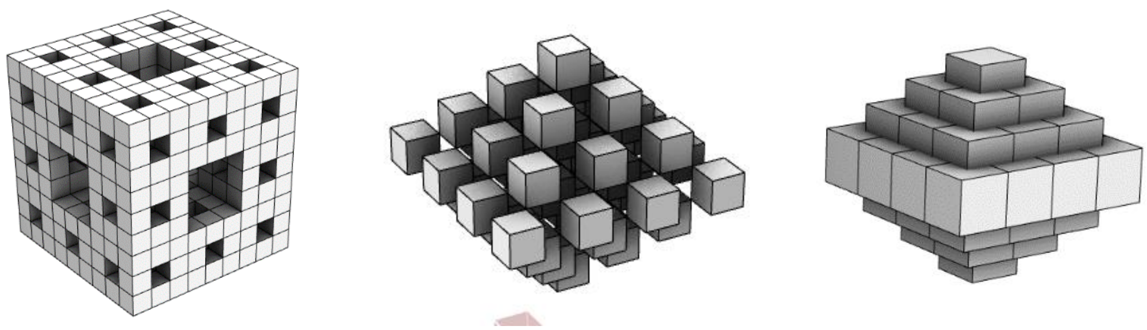


(a) (b) (c) (d)

Figure 5: *"Complement" figure with Escher's solid at level 0 (a) and 1 (b) and 2 (c) Level 1 with a type 2 distance (d).*

Thus, polyhedral that tessellate space (cube, Escher's solid, octahedron cube, e.g.) are relevant because they allow us to make figures that fit together at the appropriate distance, without leaving gaps. If we extend the distance or consider the fractal level to calculate the distance, as in the Menger figure itself, we will obtain new and different figures. And even by controlling the different parameters we can build infinite types of Menger Vector list fractals, for example: Allowing the base polyhedral overlap, making them touch each other only on one side or on one vertex... etc. In this way, a whole world of possibilities unfolds.

We use an algorithm that makes recursive figures increasingly larger. The result can be adjusted by simply adding a final scaling down. Figure 6 shows different variations of the Menger Sponge using cubes and changing distances. We can see if at each level the distance between "atoms" changes, in this way (*i**10*3**l, *j**10*3**l, *k**10*3**l) depending on the three-dimensional indices (*i, j, k*) and the level l or in the other two images it is always l=1.

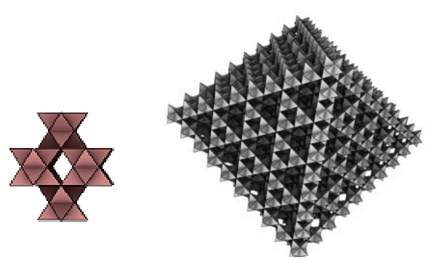


(a) (b) (c)

Figure 6: *Classical Menger Sponge (a) and its complement modifications changing the distances randomly (b and c).*

Based on these variations, it can be concluded that there are an enormous number of variables that can be "played" with. It is beyond the scope of this article to calculate the concrete number of possible variations following the established rules, but we could certainly create numerous other exploitable and aesthetically appealing figures. We should bear in mind that many unproductive rules such as the null rule (all zero) or rule 1 (all 1) should be avoided.

As a final sample, a Menger structure at level 2 is shown in Figure 7. The major peculiarity of his figure is the initial "atom", a compound of Stella Octangula, which is itself a complex figure.

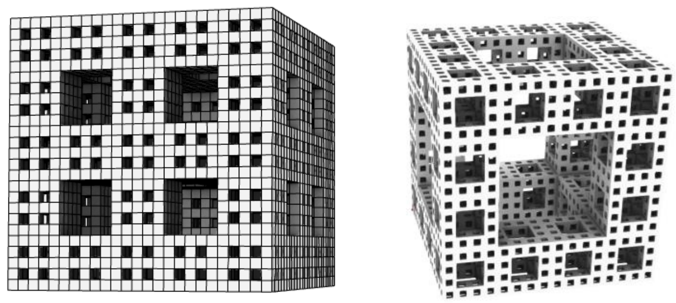


(a) (b)

Figure 7: *Menger "complement" figure at level 0 (a) and level 2 (b), using a compound of Stella as the initial 'atom'.*

The Menger-Diaz fractals

Some tangential challenges have already been raised concerning this issue, such as the Sierpinski Carpet in 2D, where the program and algorithms that model them are the same as in 3D, with the third coordinate set to 0 [5]. Others, such as The Jerusalem Cube are something more complicated [6]. Many, many other versions (we might even say infinite) can be constructed using variants of this algorithm. As an example, in Figure 8 we can see several modified Merger Sponges created by extending the vector list. They are made with a nested list of vectors 5x5 (a) and 7x7 (b). More details are explained in [2].



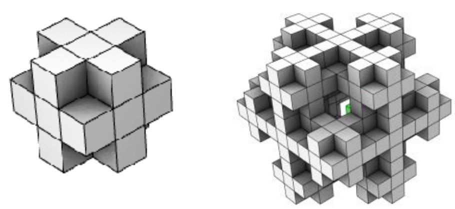
(a) (b)

Figure 8: *Modified Menger-Diaz Sponges at level 2.*

However, recently, an interesting variation called the Mosely snowflake was developed [7]. The Mosely snowflake is a cube-based fractal with corners recursively removed; therefore, we can then draw the following vector list.

a= [[[0,1,0], [1,1,1], [0,1,0]], [[1,1,1], [1,0,1], [1,1,1]], [[0,1,0], [1,1,1], [0,1,0]]](Mosely)

Consequently, we found that the algorithm covers some other examples of variations, as shown in Figure 9.



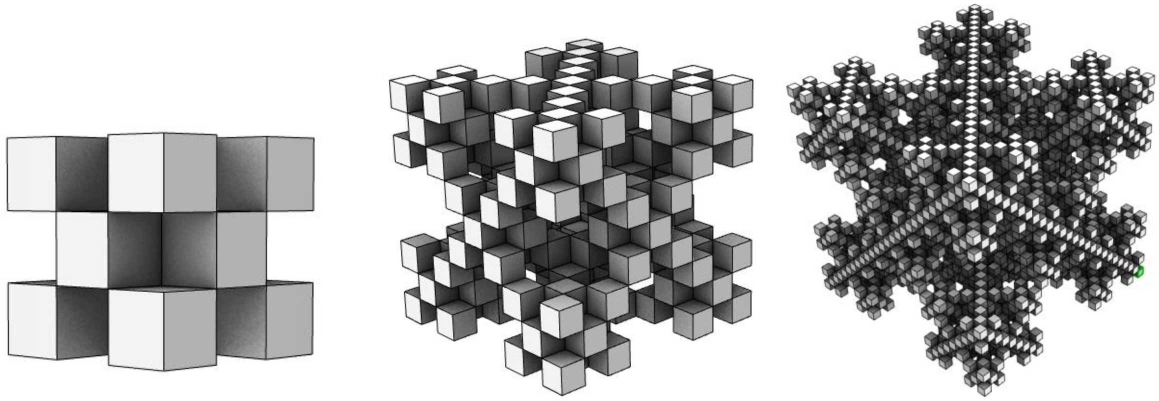
(a) (b)

Figure 9: *Mosely [7] figures at levels 1 (a) and 2 (b).*

Following some of the guidelines outlined throughout the article, we can create a wide range of variants, which from now on we will refer to as Menger-Diaz fractals. The first, is the Menger "complement". In others, the algorithm adopted is as follows:

a=[[[1,0,1],[0,1,0],[1,0,1]],[[0,1,0],[1,0,1],[0,1,0]],[[1,0,1],[0,1,0],[1,0,1]]] (*Menger-Diaz 2*)

The number of figures that can be developed from this rule is manifold. The first examples of Menger-Diaz figures are shown in Figure 10.



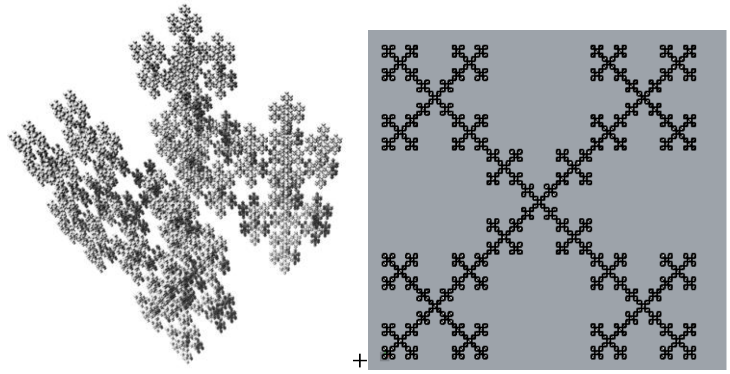
(a) (b) (c)

Figure 10: *Menger-Diaz figures at level 1 (a), level 2 (b), and level 3 (c).*

If we modify the vector lists, we can still create works of art simply and mechanically.

a=[[[1,0,1],[0,0,0],[1,0,1]],[[0,0,0],[0,1,0],[0,0,0]],[[1,0,1],[0,0,0],[1,0,1]]] (*Menger-Diaz 3*)

More possible variants, such as the one shown in Figures 11 (rule 3) and 12 (rule 4), are shown below. In Figure 11 a very familiar fractal appears when we oversample the figure into a horizontal plane.



(a) (b)

Figure 11: *Menger-Diaz figures at level 4 (a) and its horizontal plane (b).*

Now a=[[[0,1,0],[1,0,1],[0,1,0]],[[1,0,1],[0,1,0],[1,0,1]],[[0,1,0],[1,0,1],[0,1,0]]] (*Menger-Diaz 4*)

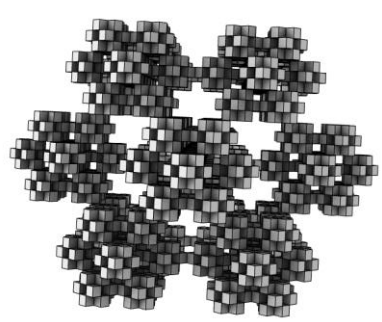


Figure 12: *Menger-Diaz figure at level 3.*

The algorithm

In PythonScript of Rhino for a Sierpinsk tetrahedron but is the same for any list of nested vectors changing the variable a and the size of the cube. For each level you must repeat the algorithm changing l

```
import rhinoscriptsyntax as rs
objs = rs.AllObjects()
l=1 # the level
d=30#distance. Its depends on the polyhedron and their size
a= [[[1,0], [0,1]], [[0,1[1,0]]]
for i in range(2):
    for j in range(2):
         for k in range(2):
              if a[i][j][k]==1:
                   t=rs.CopyObject(objs,(i*d*2**l,j*d*2**l,k*d*2**l))
rs.DeleteObjects(objs);
```

Differences between Menger and Menger-Diaz Algorithm

The traditional algorithm divides a cube into smaller cubes respecting the shape and eliminating certain subdivisions. Our algorithm does not divide the cube but places it at predetermined distances that depend on n^l, where l is the level and n is the size of the units of the cube. Enlarges rather than divides the figure (although we are supposed to eventually scale or shrink to the size of the first figure at level 0).

To understand the difference between the algorithm that we propose and Menger's, we must talk about the digital world and how in that world it is very easy to copy or scale elements. We copy elements and reposition them in various positions continuously in the digital day-to-day. Menger, in his pre-digital or analog world, considered it possible to go from top to bottom, dividing a figure by extracting its entrails, predefined parts, recursively. And for this, he uses an easily divisible figure, a cube. In our proposal we go from below (from the first brick) building the figure using Menger recursion.

We build, we do not separate, from the first brick. And, in each iteration, the figure obtained is the brick of the new step. This makes the algorithm more flexible, allowing it to have an algorithm of only a few lines, but usable in more situations. This:

1-Lay a first brick. 2-Put the brick in the loop. 3-It places, following the calculated distance, the copies of the initial shape of the loop at the predefined points of existence. 4-Convert the resulting figure to the initial figure from step 2 and repeat the loop.

So, Menger-Diaz democratizes the algorithm of Menger. It's easy. It's understandable. Everyone can use it.

In Menger's Algorithm the number 3, the cube, and 7 empty spaces are fundamental. Some daring ones that we mention in the bibliography keep the number 3, and the cubes, and change the position or number of the empty spaces. We do not have all the possible bibliography references, so we cite the well-known ones, even the recommended ones, but that we were not previously aware of.

In our algorithm we keep the cube as the main shape (so that it is better understood) and we make some images of shapes with n=3, not to show that these are our main contributions, but so that it is understood and compared with something that the reader knows, but we are talking about cubes made up of n^3 cubes where n goes from 2 to 3 (Menger) and all the numbers you want to come up with. See the preprint [2]. This is when the infinity of possible forms is triggered, since for n=6 there are 2^6^3 possible fractals, it is already more than the number of atoms in the Sun and we do not have paper in the world to represent that number of fractals (mostly ugly and horrible).

And if we do not do it in a cube? Well, it can be a rectangular prism, a stepped pyramid, or any three-dimensional network of points from which you can get the values of the placement position for each copied element in each iteration. There is a lot of work ahead for anyone who wants to do it.

Summary and Conclusions

In general, playing with the initial 'atom', levels, and distances in Menger-type figure configuration can be a powerful tool for exploring the properties and behavior of these fascinating geometric figures. By experimenting with different distance configurations, it is possible to gain a deeper understanding of Menger-type figures and their many interesting properties.

Based on this method, Menger-Diaz variations have been established. Menger-Diaz variations are a new variation of Menger-type fractals that is created by modifying the way of construction.

In the field of art, Menger-like fractal structures can be used to create visually striking and complex works of art. These structures have an infinite number of faces and an infinitely complex surface, allowing for a great deal of creativity and flexibility in the design process. Artists can use these structures as the basis for paintings, sculptures, digital art, and other media and can explore a wide range of themes and styles using these structures as a starting point.

Other consequences are in the field of classification of fractals: for example, the Sierpinski Tetrahedron is also a Menger-Diaz fractal. In this preprint article [2] we explain this and other themes.

References

[1] K. Menger, *Dimensionstheorie*. 1928.
[2] M. Diaz and L. Diaz. Variations on Menger-Díaz Fractals. https://easychair.org/publications/preprint/QFNq
[3] C. A. Reiter, "Sierpinski fractals and GCDs," *Comput Graph*, vol. 18, no. 6, pp. 885–891, Nov. 1994, doi: 10.1016/0097-8493(94)90015-9.
[4] Benoît B. Mandelbrot, *Fractals: Form, Chance, and Dimensions*. San Francisco, 1977.
[5] Y. D. Sergeyev, "Evaluating the exact infinitesimal values of area of Sierpinski's carpet and volume of Menger's sponge," *Chaos Solitons Fractals*, vol. 42, no. 5, pp. 3042–3046, Dec. 2009, doi: 10.1016/j.chaos.2009.04.013.
[6] Eric Baird, "The Jerusalem cube," *Magazine Tangente*, p. 45, 2013.
[7] L. Wade, "Folding Fractal Art from 49,000 Business Cards," *Wired*, 2017. Accessed: Dec. 28, 2022. [Online]. Available: https://www.wired.com/2012/09/folded-fractal-art-cards/
[8] George W. Hart. Procedural Generation of Sculptural Forms. https://archive.bridgesmathart.org/2008/bridges2008-209.html
[9] https://demonstrations.wolfram.com/SubstitutionSystemsIn3D/
[10] https://blogs.mathworks.com/cleve/2022/01/15/the-menger-sponge-complement-and-associate/