Circles Packing and Wallpaper Pattern

Bill Huang
Oberlin College
chuang@oberlin.edu

Abstract

Inspiring from Douglas Dunham and John Shier, in this paper I will present a detailed algorithm of filling out square with randomly placed and progressively smaller non-overlapping circles. Then, I extend the algorithm to fill out triangles and use the filled squares and triangles to tile a plane, which yields several wallpaper group patterns. These patterns reveal the beauty in the combination of global symmetry and local randomness.

Introduction

In their paper "The Art of Random Fractals", Douglas Dunham and John Shier describes a mathematically based algorithm that can fill out a region with a sequence of randomly placed and progressively smaller shape [1]. In their second paper "Fractal Wallpaper Patterns", Dunham and Shier extend the algorithm to create wallpaper pattern that are locally random but globally symmetry [2]. Inspiring from their artworks, I spend time investigating their algorithm and incorporating a geometrical perspective to develop a more detailed coding-based algorithm. In the first section of this paper, I will present a concrete version of the pseudo code that I use to fill out a square with varying size of non-overlapping circles. With the filled square, I can replicate it and also generate its reflections to tile the plane that yields locally fractal but globally symmetry pattern, such as Figure 1. In the next section, I will present some patterns I create on pmm, p3m1, p4m, and p6m wallpaper group. Last, I summarize the results and draw conclusions.

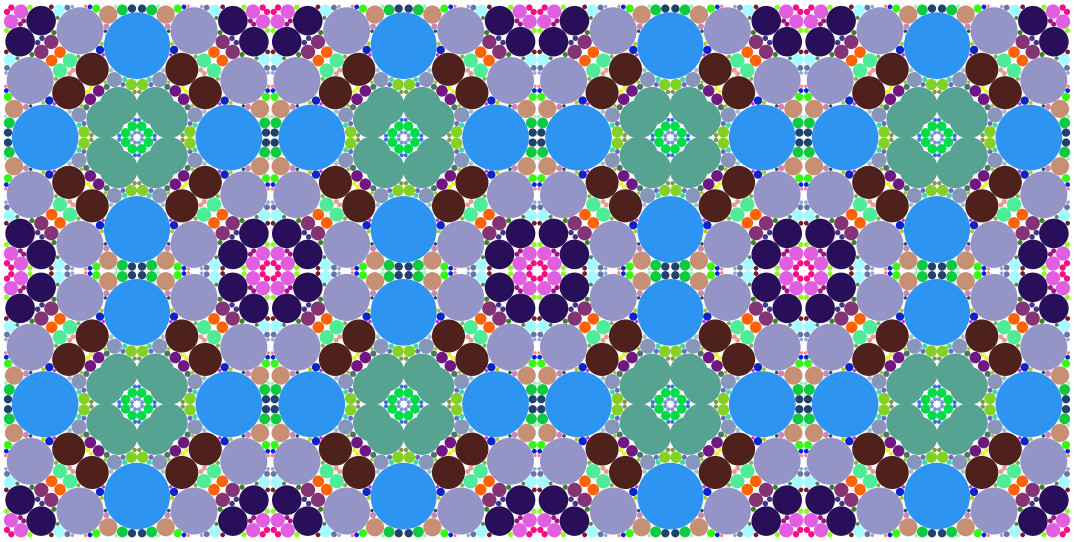

Figure 1 : *A pattern of circles with global p4m symmetry*

The Algorithm

The key idea of the algorithm is to randomly place progressively smaller non-overlapping circles into a square (or a triangle), but before we go into the detail of placing the circles, we need to first decide the size of the circles. Dunham, Shier, and Paul Bourke all suggests that the circles should obey an inverse power law area rule:

$$A_i = A_1 i^a,$$

where A_i is the area of C_i, the ith circle we place, A_1 is the area of C_1, and $a < -1$ so that $\sum_{i=1}^{\infty} A_i$ converges [1, 2, 3]. In my algorithm, I choose $a = -1.01$ to have circles decreasing slowly. After deciding the size of circles, we also need to define the region we want to fill out. Here, I will give procedure on how to fill a $4 * 4$ square starting with circle of radius 1.

The first major problem I have to deal with is where to locate the center of the circle so that it is not overlapping with previously placed circles and the boundary of the square. Before that, let me set up a coordinate system. I will take the center of the square as the origin, so the range for x and y are $-2 \leq x, y \leq 2$. Let $r_j = \sqrt{A_j/\pi}$ be the radius, where A_j is decided from the inverse power law area rule above, and (x_j, y_j) be the coordinate for C_j. Then the pseudo-code for finding the center of C_i, the ith circle, is the following:

1. Initiate a random number generator to generate x_i and y_i in the range $-2 + r_i \leq x_i, y_i \leq 2 - r_i$ so that C_i will not overlap with the boundary of the square.
2. Let $(x_1, y_1), \ldots, (x_{i-1}, y_{i-1})$ represents the coordinate for the center of $C_1, \ldots, C_{i-1}$ respectively. In order to have C_i non-overlapping with $C_1, \ldots, C_{i-1}$, we need to have

   $$\sqrt{(x_i - x_k)^2 + (y_i - y_k)^2} \geq r_i + r_k \quad \forall k \in \{1, 2, \ldots, i-1\}.$$

   In words, this means that the distance between the center of C_i and the center of C_k must be greater than the sum of their radius, and this condition must hold for all $k < i$ so that C_i is not overlapping with $C_1, \ldots, C_{i-1}$ [3].
3. If (x_i, y_i) meet the conditions above, then use (x_i, y_i) as the center for C_i. If not, then repeat the process again and again until we find a (x_i, y_i) that meet the conditions above.

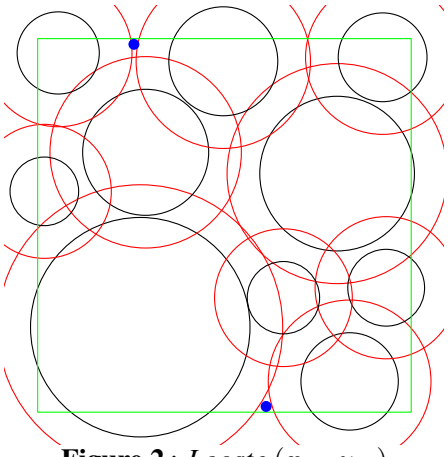

Figure 2: *Locate* (x_{11}, y_{11})

Figure 2 provides a geometrical perspective for the pseudo-code above. Here, we try to find the the center for C_{11} with radius r_{11}, where $r_{11} = \sqrt{A_{11}/\pi}$. The circles in black line represent $C_1, \ldots, C_{10}$, which C_1 is the largest circle and C_{10} is the smallest circle, and their radius are $r_1, \ldots, r_{10}$ respectively. The circles in red have radius be the sum of r_{11} and the radius of the black circles they co-center with. Then in order to have C_{11} not overlapping with $C_1, \ldots, C_{10}$, we need the center of C_{11} be a point on the boundary or outside of all the circles in red. Moreover, all the points inside the green square is

$$\{(x, y) : x \in [-2 + r_{11}, 2 - r_{11}], y \in [-2 + r_{11}, 2 - r_{11}]\},$$

and they will not overlap with the boundary of the $4 * 4$ squares. Therefore, the dots in blue are the two possible centers for C_{11} in this case.

However, since my algorithm pick x_i and y_i in random, it is possible that the computer cannot find the center for a circle we want to place. Hence, I set 500,000 as the upper limit for number of trials to find (x_i, y_i),

meaning that if we cannot find a (x_i, y_i) that can be the center of C_i in 500,000 trials, we stop looking for another (x_i, y_i), and this produce Figure 3 when I run it. Nevertheless, from Figure 3, there is still open space we can place some smaller circles. Hence, let us move to the second stage of the algorithm.

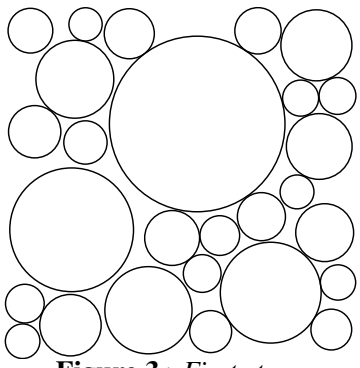

Figure 3 : *First stage*

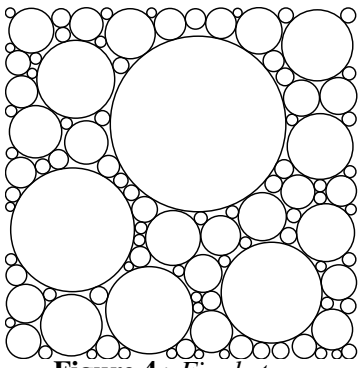

Figure 4 : *Final stage*

The core of the second stage is the discretization of the square into points and check whether these points generated can be the center of some smaller circles we can place. With that in mind, I take step size of 0.01 unit (recall that the square is $4 * 4$) to discretize the square into following points,

$$\{(x, y) : x = -2 + 0.01k, y = -2 + 0.01l \quad \text{for some integers } k, l \in 1, 2, \ldots, 400\}.$$

These give me a list of points to consider, and I will save the points that are not inside any circles generated in the first stage to the array N in (x, y) form. Then I will do the following procedures to array N for producing additional circles:

1. Let dA be the area of the circle that we cannot find the center in 500,000 trials in the first stage. Suppose we have placed n circles, then $dA = A_1(n + 1)^a$. Then the radius of C_{n+1}, the first circle we try to place in the second stage, is $dr = \sqrt{dA/\pi}$.
2. Now, run through all the points saved in array N in an order that starts from points at the bottom left to the points at the top right. Then check whether any of them can be the center of a circle that fit the following criteria:
   - has radius dr
   - not overlapping with any circles that have been placed;
   - not overlapping with the boundary of the $4 * 4$ square;

   If there is a point that meet the criterion above, then the x and y-coordinate of that point will be the center of the new circle I place. If no point in N meet the criterion, I do not place a circle with radius dr.
3. Now decrease dA in the rate that is suggested by the inverse power law area rule, which in this case, $dA = A_1(n + 2)^a$ for the area of the circle we try to place second after the first stage. Keep decreasing dA and repeating procedure 2 until we reach that $dr = \sqrt{dA/\pi} < 0.05$.

After finishing up the procedures above, it yields Figure 4, and we can see that the square now is mostly filled out by progressively smaller non-overlapping circle.

Having circles in black line all over the plane does not produce a lot of aesthetic value, then I initiate a number generator to uniformly generate three numbers between 0 and 1 to be the red, green, blue value respectively for each circles. With the square filled with color circles, I can replicate it and generate its reflection. With the original square and its reflection, we can produce different wallpaper patterns by arranging them with their rotations in a correct order. I will present several examples in the next section.

Patterns with Symmetry Group pmm

Figure 5 shows a pattern of circles with global pmm symmetry which contains perpendicular axes of reflections (horizontal and vertical axes in this case) and 180° of rotations with center at where the two reflection axes intercept [4]. Moreover, Figure 5 is constructed by the square lattice generated from my algorithm with its reflection presented in Figure 6. The length of the square lattice is 4 unit and the radius of the largest circle is 1 unit. There are total of 114 circles in each square lattice and filling up 89% of the area.

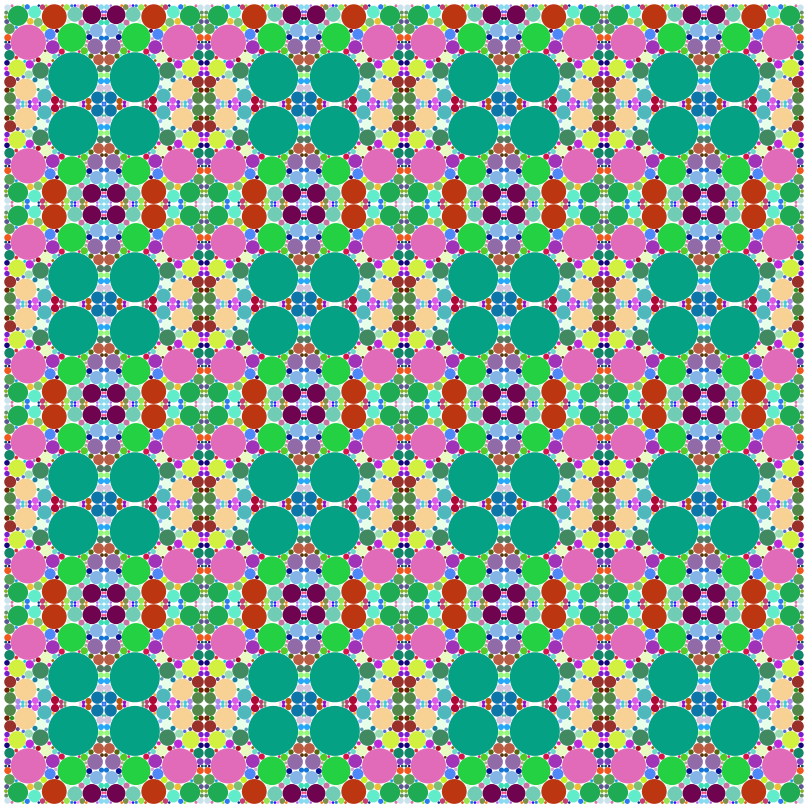

Figure 5 : *A pattern of circles with global pmm symmetry*

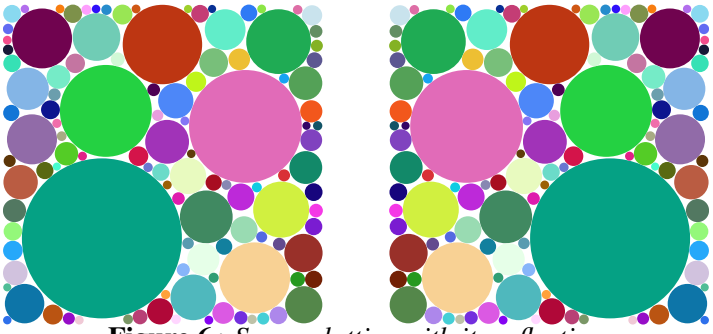

Figure 6 : *Square lattice with its reflection*

Patterns with Symmetry Group p3m1

Figure 7 shows a pattern of circles with global p3m1 symmetry which contains refection with axes inclined at 60° to one another and 120° rotations with center at the reflection axes [4]. Moreover, Figure 7 is constructed by the equilateral triangle generated from my algorithm with its reflection presented in Figure 8. The length of the triangle lattice is 6 unit and the radius of the largest circle is 1 unit. There are total of 110 circles in each equilateral triangle and filling up 90% of the area.

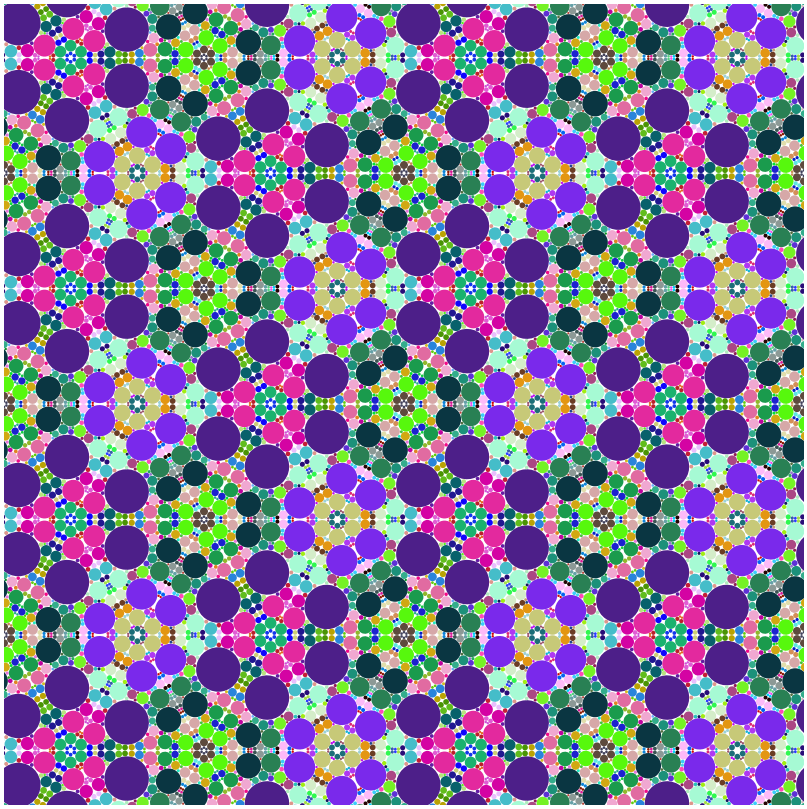

Figure 7 : *A pattern of circles with global p3m1 symmetry*

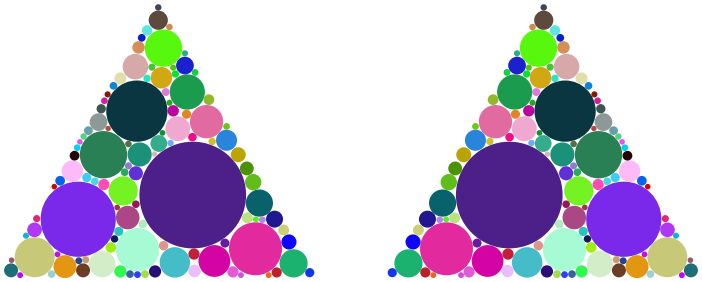

Figure 8 : *Equilateral triangle lattice with its reflection*

Patterns with Symmetry Group p4m

Figure 9 shows a pattern of circles with global p4m symmetry. The p4m symmetry group contains reflection with their four axes inclined at 45° to one another. It also has 90° and 180° rotations with their centers on the axes of reflection. There are two glide reflections as well [4]. Moreover, Figure 9 is constructed by the isosceles right triangle generated from my algorithm with its reflection presented in Figure 10. The length of the equal sides of the triangle lattice is 6 unit and the radius of the largest circle is 1 unit. There are total of 150 circles in each isosceles right triangle lattice and filling up 88% of the area.

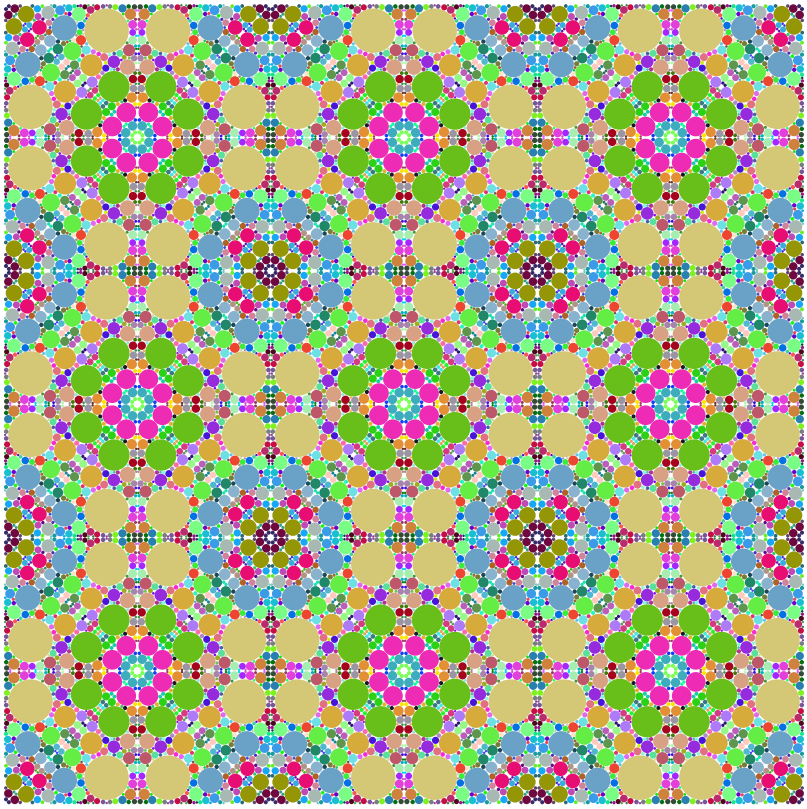

Figure 9 : *A pattern of circles with global p4m symmetry*

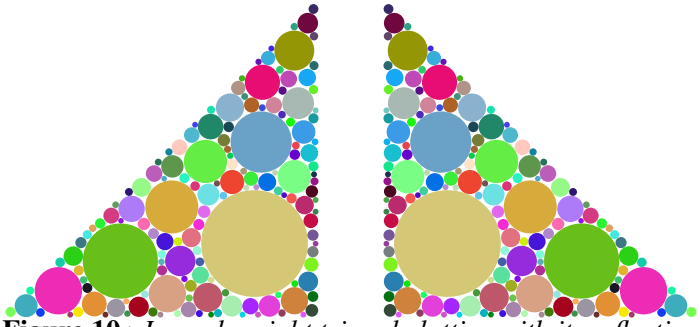

Figure 10 : *Isosceles right triangle lattice with its reflection*

Patterns with Symmetry Group p6m

Figure 11 shows a pattern of circles with global p6m symmetry. The p6m symmetry group contains reflection with their six axes inclined at 30° to one another. It also has 60°, 120°, 180° rotations with their centers on the axes of reflection [4]. Moreover, Figure 11 is constructed by the 30° - 60° - 90° triangle generated from my algorithm with its reflection presented in Figure 12. The length of the shortest side of the triangle lattice is 4 unit and the radius of the largest circle is 1 unit. There are total of 88 circles in each 30° - 60° - 90° triangle lattice and filling up 90% of the area.

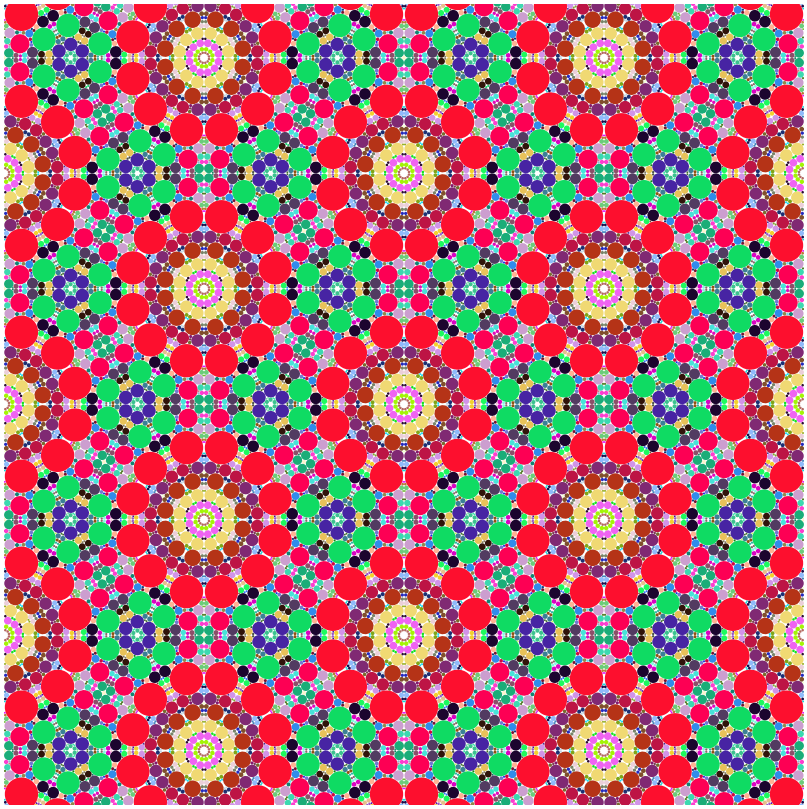

Figure 11 : *A pattern of circles with global p6m symmetry*

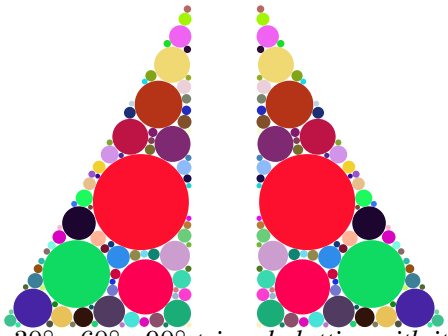

Figure 12 : 30° - 60° - 90° *triangle lattice with its reflection*

Summary and Conclusion

I have introduced an algorithm of filling squares with progressively smaller non-overlapping circles. With a similar idea, we can fill out other regular polygons with non-overlapping circles as well. Then, we can use these filled polygons, especially triangles, squares, and hexagons, to generate all 17 wallpaper groups.

However, in my program, I decide to generate the color for the circles randomly. It would be interesting to produce some wallpaper patterns that use only a selected range of color. On the other hand, my algorithm takes approximately 20 minutes to finish filling out $4 * 4$ square with radius of biggest circle be 1 and radius of smallest circle be 0.05, and it takes more than an hour if I set the radius of the smallest circle to be 0.03. Therefore, I may continue to investigate a more efficient algorithm so that I can fill out the region with some more non-overlapping circles while keeping the running time short.

References

[1] Douglas Dunham and John Shier, The Art of Random Fractals, *Proceedings of Bridges 2014: Mathematics, Music, Art, Architecture, Culture*, Phoenix, Arizona, 2014, pp.79-86. Also available at `http://archive.bridgesmathart.org/2014/bridges2014-79.html` (as of Mar. 18, 2016).

[2] Douglas Dunham and John Shier, Fractal Wallpaper Patterns, *Proceedings of Bridges 2015: Mathematics, Music, Art, Architecture, Culture*, Phoenix, Arizona, 2015, pp.183-190. Also available at `http://archive.bridgesmathart.org/2015/bridges2015-183.html` (as of Mar. 18, 2016).

[3] Paul Bourke, Random Space Filling Tiling of the Plane, `http://paulbourke.net/texture_colour/randomtile/` (as of Mar. 18, 2016).

[4] Wallpaper, *Math and the Art of MC Escher*, `http://euler.slu.edu/escher/index.php/Wallpaper_Patterns` (as of Mar. 18, 2016).