Bubble Hierarchies

Marcel Hlawatsch, Michael Burch, and Daniel Weiskopf*
Visualization Research Center (VISUS), University of Stuttgart

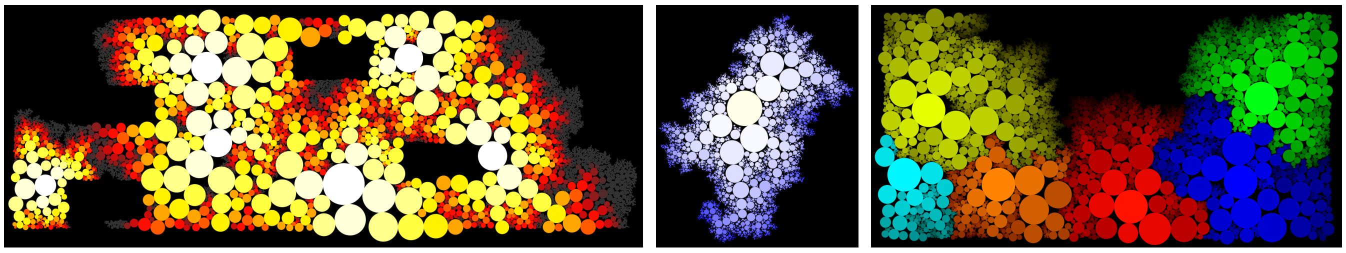

Figure 1: *Different bubble hierarchies created with our method. (a) Multiple seeds generate bubble hierarchies around several rectangular obstacles. (b) The visual look of bubble hierarchies also depends on the applied color map. (c) With multiple seeds, such merging structures can be generated. A color map with transparency is used to create an impression of depth.*

Abstract

We introduce bubble hierarchies as an approach to generating algorithmic art from random hierarchies. The technique is based on repeatedly drawing color-coded circles to illustrate parent–child relationships. The algorithm is simple and produces densely packed structures similar to the concept of Apollonian gaskets. We demonstrate the influence of different parameters on the visual outcome, such as the number of created circles or the color encoding. Our algorithm also supports multiple seeding points and obstacles that can be used to influence the layout of the hierarchy.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation

Keywords: Hierarchies, hierarchical structures, randomness, fractal structures, algorithmic art

1 Introduction

Hierarchical structures occur in a variety of forms stemming from different fields of application. In biology, e.g., all living organisms are hierarchically structured in the NCBI taxonomy [Federhen 2012]. In computing, file systems are organized in a hierarchical way. One aspect of hierarchical structures is that they can be the basis for aesthetically appealing visualizations. We focus in this paper on this aspect by visualizing randomly generated hierarchies.

Our *bubble hierarchies* are fractal-like representations for large hierarchies. Starting from an initial seed point, the hierarchy is created by connecting circular shapes to represent the relationships in the hierarchy. To achieve aesthetically appealing hierarchical structures, our proposed algorithm avoids any overlap between circles. Furthermore, it maintains a decreasing size order between circles that are in a parent–child relationship. Color coding is used as an additional visual feature that can indicate, e.g., the size of the circles or their depth in the hierarchy.

Besides describing our algorithm, we discuss and exemplify the effect of several parameters on the visual outcome, including the number of circles, the ratio of parent–child circle sizes, and color coding. Moreover, we demonstrate the effect of multiple initial seeds and the impact of obstacles on bubble hierarchy layouts.

2 Related Work

Bubble hierarchies exhibit similarities to Apollonian gaskets [Bourke 2006b]—fractal structures generated with circles. However, these fractals base on more restrictive rules and have a more symmetric and regular visual appearance than the outcome of our approach. Bubble hierarchies are related to the concept of Rapidly-Exploring Random Trees (RRTs) described by LaValle and Kuffner [2001]. For RRTs, points are set at random positions in the domain and connected to the nearest existing point. This generates a link-based hierarchical structure. Burch and Weiskopf [2013] rendered RRTs in an aesthetical way by applying a color coding depending on the depth of an RRT link in the hierarchy. Besides using circles to represent the elements of the hierarchy, our algorithmic generation exhibits a few other differences to their work. We also use points at random positions to create new elements. However, we have the restriction that the visual objects representing the elements—circles in our case—do not overlap. Furthermore, they must be smaller in size than the ones of the related parent elements. Other approaches that can be used to generate space-filling structures include methods based on the Traveling Salesman Problem [Bosch and Herman 2004; Kaplan and Bosch 2005], diffusion-limited aggregation [Bourke 2006a], or the approach by Long and Mould [2009]. These approaches can be used to algorithmically generate aesthetically appealing images. However, they are not based on hierarchical structures as our approach does.

In the field of information visualization, the visual representation of hierarchical data is an important topic. Different visual metaphors for this type of data have been developed over the years. This includes node-link diagrams [Reingold and Tilford 1981], indented plots [Burch et al. 2010] following the principle of indentation, and layered icicles [Kruskal and Landwehr 1983] exploiting the strategy of stacking. Further metaphors are treemaps [Shneiderman 1992], which use nesting layout strategies, and fractal ap-

*e-mail:{marcel.hlawatsch;michael.burch;daniel.weiskopf}@visus.uni-stuttgart.de

proaches [Mandelbrot 1982; Barneley et al. 1988; Rosindell and Harmon 2012; Beck et al. 2014]. However, all these concepts were not developed with the focus on aesthetical aspects. From a visualization perspective, our approach exhibits an important difference compared to existing node-link tree visualizations: it generates space-filling representations of hierarchical data, which is discussed for various visual metaphors for hierarchies by McGuffin and Robert [2009], while still showing representations of all hierarchy elements. This is, e.g., hard to obtain with space-filling treemap representations [Shneiderman 1992], in which the nesting concept leaves less space for displaying inner nodes. Similarly, the concept of circular treemaps [Wetzel 2004; Fischer et al. 2012] uses nested circles, but, as a consequence, the approach is less space-efficient.

3 Bubble Hierarchies

A bubble hierarchy is a set of connected circles that meets certain constraints. First, the circles do not overlap or intersect. Second, every circle must be tangent to its parent circle. Finally, the ratio of the radii of connected circles must be in a specific range. Figure 1 shows examples of bubble hierarchies created with our approach.

To create a bubble hierarchy, our method uses randomly selected points in the domain and the following rules (see also Figure 2) to create new circles:

1. The center of a new circle must not lie inside existing circles.
2. Use the closest existing circle as parent. For this, the distances to the circle edges and not to their centers are considered.
3. The radius of the new circle, which equals to the distance to the circle edge of the parent, lies inside the defined range.

If one of these rules is not fulfilled, a new iteration is started from the beginning. This is repeated until the desired number of bubbles is created. Details of the rules are discussed in the following.

To create a new circle, its center needs to be determined. We simply choose a random point in the domain and then check the first rule. If the point lies inside an existing circle, it is discarded and a new iteration starts. We can additionally extend this rule to support obstacles (see below) or special boundary shapes. Points inside an obstacle or outside the boundary shape are also discarded.

After a suitable center was found, the parent circle is determined. To create a set of non-intersecting circles, the parent circle must have the smallest distance between its edge and the determined center. If the distance to the center of the parent would be used, overlapping circles can occur if a circle with a smaller radius has a closer center than other circles with larger radii (see Figure 2(b), point D). After the parent is determined, the new circle is connected to it by assigning a radius such that both circles are tangential to each other.

However, to create specific hierarchies, the radii of new circles must also fulfill certain rules. To create, e.g., a top-down hierarchy, the radius of the new circle must be smaller than the radius of its parent circle. If this is not the case for the current combination of circle position and distance to the parent, the current position is discarded and the next iteration starts from the beginning.

Further extensions are possible. Obstacles can be introduced by testing the seed for a new circle against specific areas. We use the simple case of rectangular obstacles (see Figure 1(a)), but more complex shapes are possible. Since we only test the center of the circles, the circles can still overlap with the obstacles. This is on purpose because we want to achieve more organic looking structures. The same holds for the domain boundary: parts of the circles may be outside the domain, only the center must lie inside. It is also easy to use multiple seeds from which multiple hierarchies grow.

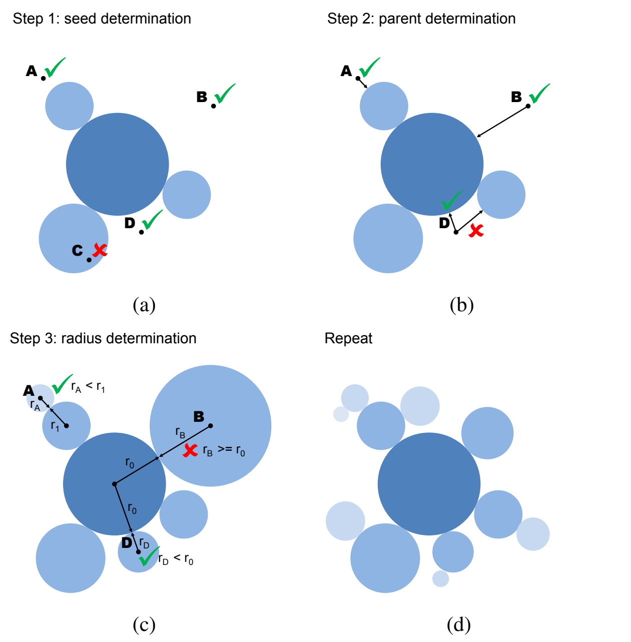

Figure 2: *Basic steps of our algorithm. (a) First, the seed position is checked. Seeds inside existing circles are discarded. (b) Next, to select the parent, the nearest circle is determined using the distance to the circle edge. (c) Now, it is checked if the radius of the new circle fulfills the rules. (d) The previous steps are repeated.*

For this, the respective number of non-intersecting circles is created initially. After creating these seeds, the algorithm is applied as in the case of a single seed. The different hierarchies seem to merge (see Figure 6), but the rule set ensures that they do not intersect.

We are aware of the fact that using totally random positions is not an efficient approach because the chance to find a suitable position for a new circle decreases with the number of already generated circles. However, we decided to use this approach to keep the algorithm as simple as possible.

4 Playing with the Parameters

In this section, we demonstrate and discuss the influence of different parameters on the visual outcome of bubble hierarchies.

Random Number Generation: A core element of our method is the random selection of seed points. Therefore, random number generation has a strong influence on the outcome. Figure 3 shows results for different initial seed numbers for the random number generator. We can see from the images that the shape of the hierarchy can vary a lot and that it is not possible to predict the outcome. The controllability of the hierarchy shape can be improved by modifying the distribution of random numbers. There are many ways to do this, but we have chosen a rather simple approach: if an additionally generated random number is below a user-defined threshold, the domain for seed point selection is restricted to a user-defined area. The other steps of the algorithm remain unmodified. Figures 3(d) and 3(e) show respective results. We can see that more circles are created in the defined area. However, this has not the effect of stretching the hierarchy in this direction, but deeper levels of the hierarchy with smaller circles are developed faster there.

Number of Circles: A basic parameter is the number of circles in the hierarchy. This is not equal to the number of iterations of the algorithm because many seed points may be discarded. Figure 4 shows that the large scale structure is already developed with a few hundred circles. Generating several thousands of circles usually adds only small details in the hierarchy.

Parent–Child Size Ratio: The ratio between the newly created child circle and its parent circle is another important parameter. It influences the growth behavior of the hierarchy (Figure 5). Allowing a fast decrease (Figure 5(a)) has typically the effect that the hierarchy requires less space. Allowing only circles with a higher ratio (Figure 5(b)) forces the hierarchy to cover a larger area of the domain. Allowing the growth of circles, i.e., a parent–child ratio larger than one, quickly fills the full domain due to an exponential growth process (Figure 5(c)). However, since we want to create a hierarchy, we do not use such ratios. By using a ratio that depends on the position in the domain, we can gain more control over the directional growth of the hierarchy. Again, we use a rather simple approach for this, although more complex adaptation methods are possible, e.g., by using mathematical functions. We simply define an area in the domain in which the ratio is larger. With this technique, we can filter parts by letting the circle sizes decay faster there (Figures 5(d) and 5(e)).

Multiple Seeds and Obstacles: Our algorithm directly supports multiple hierarchies starting at different seed positions (see Section 3). Setting multiple seed points allows a better control of the spatial distribution of circles (Figure 6). However, the detailed structure still depends on the previously discussed parameters. Increasing, e.g., the parent–child ratio (compare Figures 6(b) and 6(c)) results in a larger coverage of the domain. The images show that the impression is created that the different hierarchies are merged when using the same color coding, even though the hierarchies are actually separated. Obstacles can be used to exclude parts of the domain from being covered by the bubble hierarchy (Figures 6(d) and 6(e)). They can also be used to influence the shape of the hierarchy and force its growth into specific directions.

Color Coding: We use color to allow better perception of the hierarchy structure and to create more expressive images. Different properties can be encoded by color like circle size (Figure 7(a)) or the depth of the hierarchy (Figure 7(b)). Encoding the hierarchy depth provides a smooth color transition from the inside to the outside; we therefore use it as the default encoding. The color map can also be used to make parts invisible by applying the background color (Figure 7(c)). In the case of multiple seeds, there are additional options for applying a color map. Encoding the different hierarchies with the same color map generates the impression that they are merged (Figure 6). It is also possible to visually separate the different hierarchies with color (Figure 7(d)). Transparency can then be used to encode the depth in each hierarchy (Figure 7(e)).

5 Conclusion and Future Work

We presented bubble hierarchies as an aesthetically pleasing way to generate space-filling hierarchy representations with a simple algorithm. To illustrate the aesthetics and explore the design space, we showed the influence of different parameters on the visual outcome.

Future work could deal with accelerating our approach by parallelizing the hierarchy generation or replacing the random generation of seed points. It would also be interesting to extend the approach to 3D domains with spheres instead of circles. In this case, issues with occlusion and depth perception must be considered. Additionally, new rules and parameters could be introduced to create different structures or provide more influence on the result. We also plan to visualize real-world hierarchical data such as file systems with our approach. We see some benefit in our densely packed and space-filling representation for visualizing as much data as possible while preserving the visibility of the inherent hierarchical structure.

References

BARNELEY, M. F., DEVANEY, R. L., AND MANDELBROT, B. B. 1988. *The Science of Fractal Images*. Springer, New York.

BECK, F., BURCH, M., MUNZ, T., DI SILVESTRO, L., AND WEISKOPF, D. 2014. Generalized Pythagoras trees for visualizing hierarchies. In *Proceedings of the International Conference on Information Visualization Theory and Application*, 17–28.

BOSCH, R., AND HERMAN, A. 2004. Continuous line drawings via the traveling salesman problem. *Operations Research Letters 32*, 4, 302–303.

BOURKE, P. 2006. Constrained diffusion-limited aggregation in 3 dimensions. *Computers & Graphics 30*, 4, 646–649.

BOURKE, P. 2006. An introduction to the Apollonian fractal. *Computers & Graphics 30*, 1, 134–136.

BURCH, M., RASCHKE, M., AND WEISKOPF, D. 2010. Indented pixel tree plots. In *Proceedings of International Symposium on Visual Computing*, 338–349.

BURCH, M., ANDRIENKO, G., ANDRIENKO, N., HÖFERLIN, M., RASCHKE, M., AND WEISKOPF, D. 2013. Visual task solution strategies in tree diagrams. In *Proceedings of Pacific Visualization*, 169–176.

FEDERHEN, S. 2012. The NCBI Taxonomy database. *Nucleic Acids Research 40*, Database-Issue, 136–143.

FISCHER, F., FUCHS, J., AND MANSMANN, F. 2012. ClockMap: Enhancing circular treemaps with temporal glyphs for time-series data. In *Proceedings of the Eurographics Conference on Visualization (EuroVis 2012 Short Papers)*, 97–101.

KAPLAN, C. S., AND BOSCH, R. 2005. TSP art. In *Renaissance Banff: Bridges 2005: Mathematical Connections in Art, Music and Science*, 301–308.

KRUSKAL, J., AND LANDWEHR, J. 1983. Icicle plots: Better displays for hierarchical clustering. *The American Statistician 37*, 2, 162–168.

LAVALLE, S. M., AND KUFFNER, JR., J. J. 2001. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and Computational Robotics: New Directions*, B. R. Donald, K. M. Lynch, and D. Rus, Eds. A K Peters, 293–308.

LONG, J., AND MOULD, D. 2009. Dendritic stylization. *Visual Computer 25*, 3, 241–253.

MANDELBROT, B. 1982. *The Fractal Geometry of Nature*. W.H. Freeman and Company. New York.

MCGUFFIN, M., AND ROBERT, J. 2009. Quantifying the space-efficiency of 2D graphical representations of trees. *Information Visualization 9*, 2, 115–140.

REINGOLD, E. M., AND TILFORD, J. S. 1981. Tidier drawings of trees. *IEEE Trans. on Software Engineering 7*, 2, 223–228.

ROSINDELL, J., AND HARMON, L. 2012. OneZoom: A fractal explorer for the tree of life. *PLOS Biology 10*, 10, e1001406.

SHNEIDERMAN, B. 1992. Tree visualization with tree-maps: 2-D space-filling approach. *ACM Trans. on Graphics 11*, 1, 92–99.

WETZEL, K., 2004. Pebbles–using circular treemaps to visualize disk usage. http://lip.sourceforge.net/ctreemap.html (accessed: 06-06-2014).

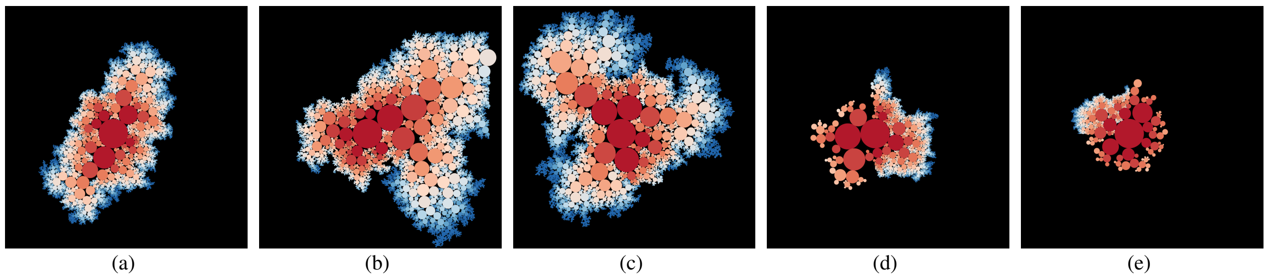

Figure 3: *Influence of random number generation. The random number generator for seed point selection was initialized differently in (a)–(c). 99% of numbers are selected only for: (d) the right half of the domain, (e) the upper left quarter of the domain.*

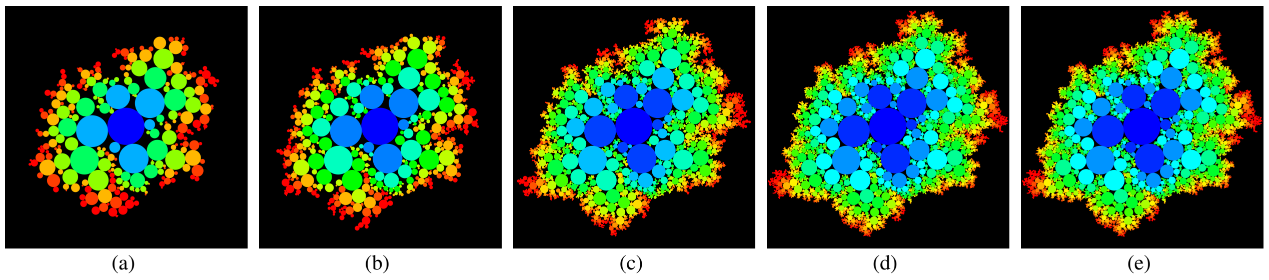

Figure 4: *Increasing number of circles: (a) 250, (b) 500, (c) 1000, (d) 2000, (e) 4000 circles were computed.*

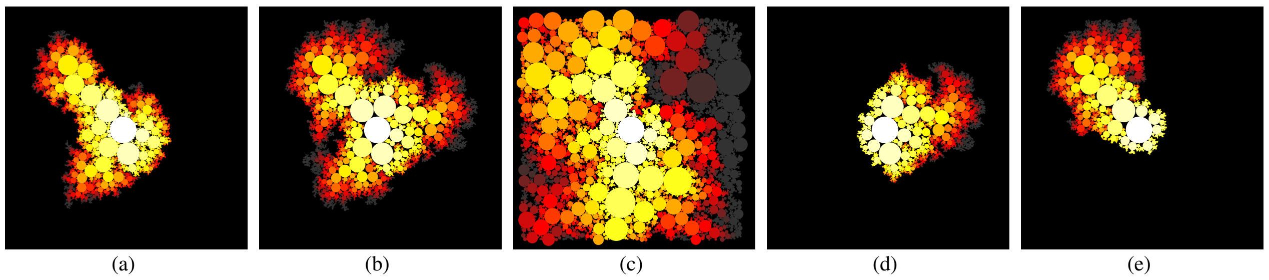

Figure 5: *Different ratios of parent and child size. (a) Child size is between 0.0 and 1.0 of parent size. (b) Child size is between 0.3 and 1.0 of parent size. (c) Child size is between 0.0 and 1.2 of parent size. The ratio can depend on the position to steer the growing direction (compare to (b)): (d) a higher ratio is allowed in the right half of the domain; (e) a higher ratio is allowed in the upper left quarter of the domain.*

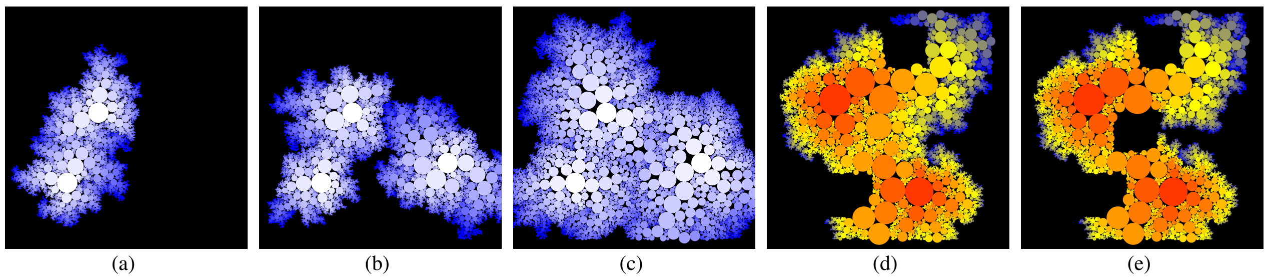

Figure 6: *Multiple initial seeds and obstacles. (a) Two initial seeding positions for 3000 bubbles. (b) Three initial seeding positions for 6000 bubbles. (c) Three initial seeding positions for 6000 bubbles with a parent–child size ratio between 0.5 and 1.0. In the other images, the ratio was between 0.25 and 1.0. Two initial seeding positions combined with (d) two and (e) three square-shaped obstacles.*

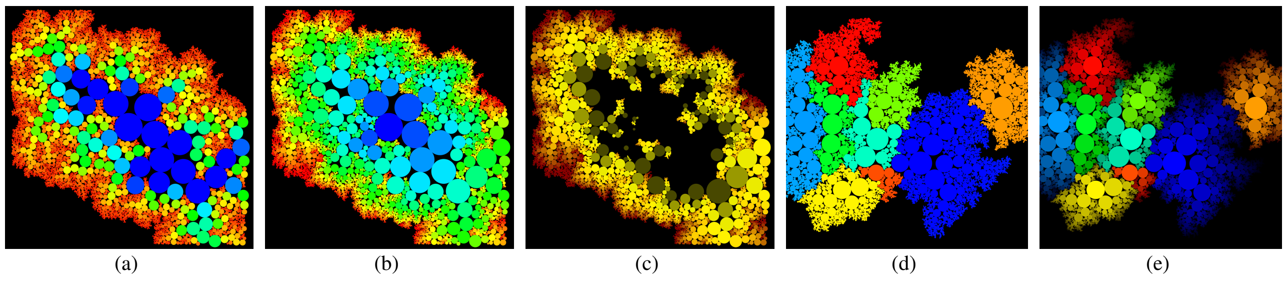

Figure 7: *Different color maps. (a) Color encoding of circle size. (b) Color encoding of hierarchy depth. (c) Masking of the inner part of the hierarchy. (d) Hierarchies from multiple seeds are colored separately. (e) Hierarchy depth is additionally encoded with transparency.*