Poster Abstracts at Eurographics/ IEEE-VGTC Symposium on Visualization (2010)
D. Auber, G. Melançon, T. Munzner, and D. Weiskopf (Chairs)

Animated, Dynamic Voronoi Treemaps

Danyel Fisher[1] and Avneesh Sud[2]

[1] Microsoft Research Redmond [2] Microsoft Research XCG

Abstract

The Voronoi Treemap is a space-filling treemap technique that relaxes the constraints of rectangular nodes. Its organic shapes maintain a one-to-one aspect ratio, are flexible with their placement, allowing stable zooming and dynamic data values. This poster articulates the tradeoffs that go into generating dynamic treemaps. We then present innovative features that allow Voronoi Treemaps to smoothly represent dynamically-changing data. We demonstrate its use with a deep organizational chart and an animated dataset. User tests confirm that Voronoi Treemaps can be a successful visualization technique.

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Animations

1. Introduction

Treemaps are a popular tool for visualizing large amounts of data. They are one of the most successful sophisticated visualizations available today: familiar examples include treemaps that visualize disk usage and stock prices. Increasingly, however, online data has gained a temporal component: we are interested in viewing streaming data, and tracking how it changes. Treemaps are well-posed to handle large data if they can portray changing values. Infovis research has attempted to solve this problem with little success.

Most space-filling treemap algorithms today depend on greedy algorithms that fill the space with rectangles. While easy to compute and quick to render, these force a choice between stability in the face of dynamic data and a low aspect ratio. A low aspect ratio is important because it can be hard to estimate the relative size of highly imbalanced shapes; stability is important in order to ensure that animation will be smooth from frame to frame, without large changes in objects. Techniques like "‘slice and dice"’ allow nodes to change size rapidly and smoothly, but its nodes tend to be long and skinny; squarified treemaps maintain a near-square aspect ratio at the cost of stability. Table 1 compares the tradeoffs of popular treemap algorithms.

Figure 1: *Stable animation at three successive monthly time points based on marketing data. These frames were extracted from a video spanning nine months.*

Voronoi treemaps [BD05, BDL05] are generated with an optimization algorithm that does not produce rectangles, which allows them to maintain both dynamic stability and a desirable aspect ratio. The algorithm produces multi-sided, curved shapes, not unlike soap bubbles. The cost of these technologies is computation time: past algorithms have been very slow.

We have implemented the Voronoi treemap algorithm using GPGPU techniques, allowing faster computation, visualization and rendering. We divide visualization into an initial, slow computation phase and a fast render phase. This allows us to show treemaps whose node sizes change smoothly over time. In addition, the treemap does not need to be recomputed on zoom: the placement is stable. This means that the entire tree can be pictured as a single map into which a user can zoom. Our implementation is sufficiently fast to allow real-time panning, zooming, and playback of dynamic treemaps, after an initial computation phase.

© The Eurographics Association 2010.

Figure 2: *Stable zooming on a Vornooi treemap of 120,000 nodes. Once the visualization is zoomed in, we refine one additional layer.*

1.1. GPU Computation and Rendering

Voronoi Treemaps are slow to compute because they require repeated computation of weighted Voronoi diagrams. These need to be repeatedly re-generated in an iterative process: a set of centroids is attempted, and then the diagram is updated until convergence. The work of [SFL10] shows how to compute a Voronoi treemap on a GPU. We have optimized the algorithm by running the GPU at an optimal resolution, computing as small a region as possible, and by accelerating convergence of the iterative step. Our optimizations allow a two-order of magnitude speedup: on a single 2.4 Ghz Core2 CPU and Nvidia GT260 GPU, we compute a tree of 1100 nodes in 2.5 seconds, and a tree of 11100 nodes in 24 s. (In contrast, [BD05] renders a tree of 4000 nodes in 7 minutes on 8 2.4Ghz Xeon CPUs.)

Once computation is complete, we save out a serialized description of the Voronoi tree for interactive rendering and navigation. This description is very small, and can be rendered much more quickly: no iterations are necessary, and so our GPU-based algorithm can render 1111 nodes in 23 ms. For deeper trees, we limit the number of layers we render; as the user zooms, we show more nodes (Figure 2).

2. Stable Hierarchical Updates

Voronoi treemaps are amenable to smooth changes of the data–that is, the visualization changes smoothly as the data does. This is possible because cells can grow in multiple directions, and can slide relative to each other. Cells can be initially seeded with starting locations; they will tend to stay in the same place between iterations. We divide animation into two stages: data-based interpolation (in which we change data values on the tree) and visual interpolation (in which we smoothly merge nodes together). The data-based interpolation phase happens during computation, while visual interpolation occurs during rendering, and thus can be sped or slowed as needed. In between data stages, we re-initialize cells at their locations of the previous iteration, in order to minimize movement. Figure 1 illustrates a dataset changing over three time periods.

Table 1: *Trade-offs of treemap algorithms. Optimally, algorithms maintain a small aspect ratio and allow stable updates and zooming without distortion*

	Avg Aspect Ratio [TS07]	Stable updates	Stable zoom
Slice&Dice	100+	✓	X
Squarified	1.3	X	✓
Ordered	2.8	X	X
Strip	2.5	X	X
Spiral	2.5	X	X
Voronoi	*1.3**	✓	✓

* Based on the aspect ratio of the bounding boxes of the Voronoi regions. Algorithms in [Shn92, BSW02, TS07, BHvW00]

3. Conclusion

Dynamic Voronoi treemaps address a long-standing issue in treemap visualization: their data values can be updated smoothly, while they maintain a good aspect ratio. This opens future opportunities to visualize hierarchical, streaming data. Our demo and video illustrate these techniques in detail.

References

[BD05] BALZER M., DEUSSEN O.: Voronoi treemaps. In *IEEE InfoVis* (Los Alamitos, CA, USA, 2005), IEEE Computer Society. 1, 2

[BDL05] BALZER M., DEUSSEN O., LEWERENTZ C.: Voronoi treemaps for the visualization of software metrics. In *ACM SoftVis '05* (New York, NY, USA, 2005), ACM, pp. 165–172. 1

[BHvW00] BRULS M., HUIZING K., VAN WIJK J.: Squarified treemaps. In *Proc. of Joint Eurographics and IEEE TCVG Symp. on Visualization (TCVG 2000)* (2000), IEEE Press, pp. 33–42. 2

[BSW02] BEDERSON B. B., SHNEIDERMAN B., WATTENBERG M.: Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. *ACM Trans. Graph. 21*, 4 (2002), 833–854. 2

[SFL10] SUD A., FISHER D., LEE H.-P.: *Fast Dynamic Voronoi Treemaps*. Tech. Rep. MSR-TR-2010-37, Microsoft Research, 2010. 2

[Shn92] SHNEIDERMAN B.: Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph. 11*, 1 (1992), 92–99. 2

[TS07] TU Y., SHEN H.-W.: Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1286–1293. 2

© The Eurographics Association 2010.