# Multi-way Space Partitioning Trees

(Abstract)

CHRISTIAN A. DUNCAN
Department of Computer Science
Unversity of Miami
duncan@cs.miami.edu

## 1 Introduction

Problems in geometry often involve processing sets of objects in the plane or in a higher dimensional space. One very common technique used to deal with these problems is to recursively partition the space into subspaces. Within that category, the most well-known partitioning scheme is to partition the regions via a hyperplane cut. The resulting tree created is known as a binary space partitioning (BSP) tree.

The operation is quite straightforward. We take the initial input and determine in some manner a hyperplane that divides the region. We then partition the region into two subregions, corresponding to its intersection with the two half-spaces defined by the hyperplane. The set of objects are also partitioned by the hyperplane sometimes fragmenting individual objects. The process is then repeated for each subregion and the set of (fragmented) objects until each subregion (cell) contains only one fragment of an object. This requires the assumption that the objects are disjoint; otherwise, we cannot guarantee that every cell contains only one fragment of an object. The final tree represents a partitioning of the space into cells. Each node of the tree stores the hyperplane splitting that region, and each leaf represents a cell in the partitioning containing at most one fragmented object. For more detailed information see, for example, [9].

BSP trees and their variations have uses in a wide-range of applications. Fuchs et al. [12] popularized the use of BSP trees by applying them to the *painter's algorithm*, see [11], used in computer graphics. Other graphics applications include shadow generation [4, 5], solid modeling [13, 15, 19], visibility [3, 17, 18], and ray tracing [14].

The size of the constructed BSP tree, which is usually bounded by the number of times each object (line segment) is partitioned (split) by a cut, plays a key role in the performance of applications using the tree. Until recently, it was a long-standing open problem if the two-dimensional version of the BSP tree could be constructed in $O(n)$ space where $n$ is the initial number of line segments. Recently, Tóth [20] proved that, in fact, it is not possible and that there exist sets of line segments in the plane for which any BSP tree must have at least $\Omega(n \log n / \log \log n)$ size. It is well known that there exist constructions requiring at most $O(n \log n)$ nodes in the tree, e.g. [15]. Thus, the bound is almost tight.

### 1.1 Brief History

The literature on BSP trees is quite rich, and so we do not make any attempt in this abstract to provide a comprehensive list on its vast history. Paterson and Yao [15] show some of the first efficient bounds on the size of the binary space partition tree. In particular, they show that a BSP tree of size $O(n \log n)$ can be constructed in the plane and an $O(n^2)$-sized tree can be constructed in $\mathbb{R}^3$ which they prove to be optimal in the worst-case. By restricting the set of line segments, other results have been developed, e.g. see [6, 10, 16, 21]. For example, Paterson and Yao [16] prove that a linear-size BSP tree exists when the objects are orthogonal line segments in the plane. Tóth [21] shows a bound of $O(kn)$ when the number of distinct line segment orientations is $k$. In [6], de Berg et al. show that in the plane a linear size BSP tree exists on sets of fat objects, on sets of line segments where the ratio between the longest and shortest segment is bounded by a constant, and on sets of homothetic objects, that is objects of identical shape but of varying sizes. Our approach is very similar to theirs but with a different aim.

The research in higher-dimensional space is also quite rich but is not the focus of this paper [1, 2, 7, 8, 10, 15, 16].

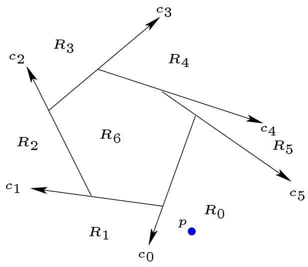


Figure 1: An example of a clockwise spiral cut $C = \{c_0, c_1, c_2, c_3, c_4, c_5\}$. Observe the 5 arm regions and the center region formed. Observe that the point $p \in R_0$ lies to the left of $c_0$ and $c_1$ but to the right of all other rays.

### 1.2 Our Results

We take a different approach that abandons the simple notion of partitioning space using a *single* hyperplane. As with the above mentioned results, our paper focuses on partitioning a set of $n$ disjoint line segments in the plane. We introduce a new data structure, the multi-way space partitioning (MSP) tree. Unlike standard binary partitioning schemes, MSP trees are produced by partitioning regions into several sub-regions using a spirally shaped cut. [1] Figure 1 shows an illustration of one such cut.

By proving that no line segment is ever pierced more than a constant number of times, we show that for *any* set of disjoint line segments in the plane there exists a linear-size MSP tree on the set. In addition, the simple nature of the spiral cuts makes it easy to traverse the tree in nearly the same manner as a regular BSP tree. Unlike previous results on linear-size BSP trees, we place no constraints on our segments other than being disjoint. Since our structure is a deviation from the standard BSP tree construction, we also describe an application of our algorithm. More specifically, we prove that the painter's algorithm can quite easily be adapted to use our structure to run in $O(n)$ time. It should also be noted that the constant factors in our algorithm are quite small. In fact, we show that the constructed tree has size less than $4n$.

As a spiral cut can have any number of spokes extending from it, the degree of any node in the tree could be linear. It is still open whether the technique can be adapted to yield trees of bounded degrees. However, we feel the more important issue with this structure is extending the idea to three dimensions and higher.

## References


[1] P. Agarwal, T. Murali, and J. Vitter. Practical techniques for constructing binary space partitions for orthogonal rectangles. In *Proceedings of the 13th Annual Symposium on Computational Geometry (SCG-97)*, pages 382–384, New York, June 4–6 1997. ACM Press.

[2] P. K. Agarwal, E. F. Grove, T. M. Murali, and J. S. Vitter. Binary space partitions for fat rectangles. *SIAM Journal on Computing*, 29(5):1422–1448, Oct. 2000.

[3] J. M. Airey. *Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision and Potentially Visible Set Calculations.* PhD thesis, Dept. of CS, U. of North Carolina, July 1990. TR90-027.

[4] N. Chin and S. Feiner. Near real-time shadow generation using BSP trees. *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24(4):99–106, Aug. 1990.

[5] N. Chin and S. Feiner. Fast object-precision shadow generation for areal light sources using BSP trees. In D. Zeltzer, editor, *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, volume 25, pages 21–30, Mar. 1992.

[6] de Berg, de Groot, and Overmars. New results on binary space partitions in the plane. *CGTA: Computational Geometry: Theory and Applications*, 8, 1997.

[7] M. de Berg. Linear size binary space partitions for fat objects. In P. G. Spirakis, editor, *Algorithms—ESA '95, Third Annual European Symposium*, volume 979 of *Lecture Notes in Computer Science*, pages 252–263, Corfu, Greece, 25–27 Sept. 1995. Springer.

[8] M. de Berg and M. de Groot. Binary space partitions for sets of cubes. In *Abstracts 10th European Workshop Comput. Geom.*, pages 84–88, 1994.

[9] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications.* Springer-Verlag, Berlin Heidelberg, 1997.

[10] A. Dumitrescu, J. S. G. Mitchell, and M. Sharir. Binary space partitions for axis-parallel segments, rectangles, and hyperrectangles. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pages 141–150. ACM Press, 2001.


[1] See the accompanying draft for details.

[11] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, Reading, MA, 1990.

[12] H. Fuchs, Z. M. Kedem, and B. Naylor. On visible surface generation by a priori tree structures. *Comput. Graph.*, 14(3):124–133, 1980. Proc. SIGGRAPH '80.

[13] B. Naylor, J. A. Amanatides, and W. Thibault. Merging BSP trees yields polyhedral set operations. *Comput. Graph.*, 24(4):115–124, Aug. 1990. Proc. SIGGRAPH '90.

[14] B. Naylor and W. Thibault. Application of BSP trees to ray-tracing and CGS evaluation. Technical Report GIT-ICS 86/03, Georgia Institute of Tech., School of Information and Computer Science, Feb. 1986.

[15] M. S. Paterson and F. F. Yao. Efficient binary space partitions for hidden-surface removal and solid modeling. *Discrete Comput. Geom.*, 5:485–503, 1990.

[16] M. S. Paterson and F. F. Yao. Optimal binary space partitions for orthogonal objects. *J. Algorithms*, 13:99–113, 1992.

[17] S. J. Teller. *Visibility Computations in Densely Occluded Polyhedral Environments*. PhD thesis, Dept. of Computer Science, University of California, Berkeley, 1992.

[18] S. J. Teller and C. H. Séquin. Visibility preprocessing for interactive walkthroughs. *Comput. Graph.*, 25(4):61–69, July 1991. Proc. SIGGRAPH '91.

[19] W. C. Thibault and B. F. Naylor. Set operations on polyhedra using binary space partitioning trees. *Comput. Graph.*, 21(4):153–162, 1987. Proc. SIGGRAPH '87.

[20] C. D. Tóth. A note on binary plane partitions. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pages 151–156. ACM Press, 2001.

[21] C. D. Tóth. Binary space partitions for line segments with a limited number of directions. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 465–471. ACM Press, 2002.