Appears in *Proceedings of the 29th IEEE International Conference on Data Engineering*,
Brisbane, Australia, April 2013, pp. 1254-1257.

Sorting in Space: Multidimensional, Spatial, and Metric Data Structures for Applications in Spatial Databases, Geographic Information Systems (GIS), and Location-Based Services

Hanan Samet
Center for Automation Research, Institute for Advanced Computer Studies
Department of Computer Science, University of Maryland
College Park, MD 20742 USA
hjs@cs.umd.edu

***Abstract*— Techniques for representing multidimensional, spatial, and metric data for applications in spatial databases, geographic information systems (GIS), and location-based services are reviewed. This includes both geometric and textual representations of spatial data.**

I. Introduction

The representation of multidimensional, spatial, and metric data is an important issue in applications of spatial database, geographic information systems (GIS), and location-based services. This is in part a direct result of the increasing popularity of web-based services such as Microsoft Bing Maps and Google Maps and Earth, as well as their deployment on gesturing-based devices such as smartphones and tablets which have also brought Apple into the picture [68]. This popularity has led to an increase in the awareness of the importance of location as an attribute in a database. The existence of the database means that the data stored therein must be retrieved and this involves searching. The efficiency of searching is dependent on the extent to which the underlying data is sorted. The conventional definition of the term *sort* is that it is a verb meaning: (1) To put in a certain place or rank according to kind, class, or nature. (2) To arrange according to characteristics. The sorting is encapsulated by the data structure used to represent the spatial data thereby making it more accessible. In fact, the term *access structure* or *index* is often used as an alternative to the term *data structure* in order to emphasize the importance of the connection to sorting.

Notwithstanding the above definition, sorting usually implies the existence of an ordering. Orderings are fine for one-dimensional data. For example, in the case of individuals we can sort them by their weight, and given an individual such as Bill, we can use the ordering to find the person closest in weight to Bill. Similarly, we can use the same ordering to also find the person closest in weight to John. Unfortunately, in two dimensions and higher, such a solution does not always work. In particular, suppose we sort all of the cities in the US by their distance from Chicago. This is fine for finding the closest city to Chicago, say with population greater than 200,000. However, we cannot use the same ordering to find the closest city to New York, say with population greater than 200,000, without resorting the cities.

The problem is that for two dimensions and higher, the notion of an ordering does not exist unless a dominance relation holds (e.g., [44])—that is, a point $a = \{a_i | 1 \leq i \leq d\}$ is said to dominate a point $b = \{b_i | 1 \leq i \leq d\}$ if $a_i \leq b_i, 1 \leq i \leq d$. Thus the only way to ensure that an ordering exists is to linearize the data as can be done, for example, using a space-filling curve (e.g., [47], [64]). The problem with such an approach is that the ordering is explicit. Instead, what is needed is an implicit ordering so that we do not need to resort the data when, for example in our sample query, the reference point for the query changes (e.g., from Chicago to New York). Such an ordering is a natural byproduct when we sort objects by spatial occupancy, and is the subject of this paper.

II. Methods Based on Spatial Occupancy

The indexing methods that are based on sorting the spatial objects by spatial occupancy essentially decompose the underlying space from which the data is drawn into regions called *buckets* in the spirit of classical hashing methods. The difference is that the spatial indexing methods preserve order. In other words, objects in close proximity should be in the same bucket or at least in buckets that are close to each other in the sense of the order in which they would be accessed (i.e., retrieved from secondary storage in case of a false hit, etc.).

There are two principal methods of representing spatial data. The first is to use an object hierarchy that initially aggregates objects into groups based on their spatial proximity and then uses proximity to further aggregate the groups thereby forming a hierarchy. Note that the object hierarchy is not unique as it depends on the manner in which the objects were aggregated to form the hierarchy. Queries are facilitated by also associating a minimum bounding box with each object and group of objects as this enables a quick way to test if a point can possibly lie within the area spanned by the object or group of objects. A negative answer means that no further processing is required for the object or group, while a positive answer means that further tests must be performed. Thus the minimum bounding box serves to avoid wasting work. Data structures such as the R-tree [16] and the R*-tree [6] illustrate the use of this method.

The drawback of the object hierarchy approach is that from the perspective of a space decomposition method, the resulting hierarchy of bounding boxes leads to a non-disjoint decomposition of the underlying space. This means that if a search fails to find an object in one path starting at the root, then it is not necessarily the case that the object will not be found in another path starting at the root.

The second method is based on a recursive decomposition of the underlying space into disjoint blocks so that a subset of the objects are associated with each block. There are several ways to proceed. The first is to simply redefine the decomposition and aggregation associated with the object hierarchy method so that the minimum bounding rectangles are decomposed into disjoint rectangles, thereby also implicitly partitioning the underlying objects that they bound. In this case, the partition of the underlying space is heavily dependent on the data and is said to be at arbitrary positions. The k-d-B-tree [46] and the R^+-tree [88] are examples of such an approach.

The second way is to partition the underlying space at fixed positions so that all resulting cells are of uniform size, which is the case when using the uniform grid (e.g., [29]), also the standard indexing method for maps. The drawback of the uniform grid is the possibility of a large number of empty or sparsely-filled cells when the objects are not uniformly distributed. This is resolved by making use of a variable resolution representation such as one of the quadtree variants (e.g., [64]) where the subset of the objects that are associated with the blocks are defined by placing an upper bound on the number of objects that can be associated with each block (termed a *stopping condition* for the recursive decomposition process) and also often referred to as a *bucket capacity*. In this case we can say that the objects are sorted into cells which act like bins (i.e., buckets). The PR quadtree [43], [62] and its bucket variants are examples of such a structure for points, while the PM quadtree family [21], [37], [72], [79] (see also the related PMR quadtree [19], [40], [41]) is an example of a variable resolution representation for collections of straight line segment objects such as those found in polygonal subdivisions as well as higher dimensions (e.g., faces of three-dimensional objects as in the PM octree [5]). An alternative, as exemplified by the PK-tree [63], [97], makes use of a lower bound on the number of objects that can be associated with each block (termed an *instantiation* or *aggregation* threshold).

Quadtrees [24], [28] and their three-dimensional octree analogs [23], [39]. have also been used widely for representing and operating on region data in two and three dimensions, respectively (e.g., [59]). In particular, algorithms have been devised for converting between them and numerous representations such as binary arrays [48], boundary codes [14], [49], [78], rasters [50], [56], [89], medial axis transforms [55], [57], terrain models [91], boundary models [92], constructive solid geometry (CSG) [73], as well as operations such as connected component labeling [52], [75], [76], perimeters [51], [74], distance [53], image dilation [1], computing Euler numbers [13], and ray tracing [60]. Many of these operations are implemented by traversing the actual quadtrees/octrees and performing the operation on each node and its neighbors [31], [54], [58], [60], [71]. Quadtrees and their variants are to be distinguished from pyramids (e.g., [93]) which are multiresolution data structures useful in spatial data mining [2].

The principal drawback of the disjoint method is that when the objects have extent (e.g., line segments, rectangles, and any other non-point objects), then an object may be associated with more than one block. This means that queries such as those that seek the length of all objects in a particular spatial region will have to remove duplicate objects before reporting the total length. Nevertheless, methods have been developed that avoid these duplicates by making use of the geometry of the type of the data that is being represented (e.g., [3], [4], [12]). Note that the result of constraining the positions of the partitions means that there is a limit on the possible sizes of the resulting cells (e.g., a power of 2 in the case of a quadtree variant). However, this means that the underlying representation is good for operations between two different data sets (e.g., a spatial join [22], [25], [26]) as their representations are in registration (i.e., it is easy to correlate occupied and unoccupied space in the two data sets, which is not easy when the positions of the partitions are not constrained as is the case with methods rooted in representations based an object hierarchy even though the resulting decomposition of the underlying space is disjoint). For an empirical comparison of these representations with respect to multidimensional point data, see [27].

III. Future Trends

In this paper, the discussion has been in the context of the traditional explicit specification geometric representation of spatial data (e.g., as latitude-longitude pairs of numbers). This is often cumbersome as users don't always think of a location in this way, and often don't know it in this way or have easy access to it, and, more importantly, are not accustomed to communicate it to others in this way. Instead, they are accustomed to specify a location textually (including verbally). A textual specification has a number of advantages. The first is that it is easy to communicate especially on smartphone devices where a textual (also increasingly verbal via speech recognition such as Siri on the Apple platform) input capability is always present. Another important advantage is that the text acts like a polymorphic type in the sense that one size fits all. In particular, depending on the application which makes use of this information, a term such as "Washington" can be interpreted both as a point or as an area, and the user need not be concerned with this question. The drawback of the textual specification of location data is that it is ambiguous. In particular, there are many possible locations named "Washington" and they must be resolved (i.e., "toponym resolution") [33], [35], [45]. Moreover, in some cases we are not even sure that the term "Washington" denotes a location as it could be a reference to the name of a person (i.e., "toponym recognition") [32]. This can be the case when processing documents such as newspaper articles [34], [67], [77], [96], tweets [86], blogs, etc. Being able to handle such specifications enables the development of map query interfaces to a wide range of spatially-referenced data thereby enabling the development of new applications such as disease tracking [30] as well as the hidden web [36]. Moreover, such interfaces enable the search to make use of spatial synonyms which result in nearest neighbor computation where the results are names of the neighbors rather than their coordinate values.

IV. Concluding Remarks

Sorting spatial and metric data is particularly useful for proximity queries usually where proximity is measured in terms of as "the crow flies" (e.g., [17], [18], [65]). However, these representations can also be used to support proximity in a graph such as a road network (e.g., [70], [80], [81], [82], [83], [84], [85]). They can also be used with different metrics such as a Hausdorff distance [42].

Interestingly, methods analogous to those that we described have also been used in cases where the only information that we have available is a distance function that indicates the degree of similarity (or dis-similarity) between all pairs of the N objects. Usually the distance function d is required to obey the triangle inequality, be non-negative, and be symmetric, in which case it is known as a *metric* and also referred to as a *distance metric*. Given a distance function, we usually partition and index the objects with respect to their distance from a few selected objects. There are two basic partitioning schemes: *ball partitioning* and *generalized hyperplane partitioning* [20]. In ball partitioning, the data set is partitioned based on distances from one distinguished object, into the subset that is inside and the subset that is outside a ball around the object In generalized hyperplane partitioning, two distinguished objects p_1 and p_2 are chosen and the data set is partitioned into two sets based on which of the two distinguished objects is the closest. It is interesting to observe that both schemes achieve a partitioning of the underlying data set into spatial-like zones. However, the difference is that the boundaries of the zones are more well-defined in the case of ball partitioning methods as they can be expressed explicitly using a small number of objects and a known distance value. In contrast, in the case of generalized hyperplane partitioning methods, the boundaries of the zones are usually expressed implicitly in terms of the distinguished objects, instead of explicitly, which may require quite a bit of computation to determine. In fact, very often, the boundaries cannot be expressed explicitly as, for example, in the case of an arbitrary metric space (in contrast to a Euclidean space) where we do not have a direct representation of the 'generalized hyperplane' that separates the two partitions.

The functioning of the various spatial sorting methods can be experienced by trying VASCO [7], [8], [9], [11], a system for Visualizing and Animating Spatial Constructs and Operations. VASCO consists of a set of spatial index JAVA$^{\mathrm{TM}}$ applets that enable users on the worldwide web to experiment with a number of hierarchical representations (e.g., [61], [62], [64]) for different spatial data types, and see animations of how they support a number of search queries (e.g., nearest neighbor and range queries). The VASCO system can be found at `http://www.cs.umd.edu/~hjs/quadtree/`. For an example of their use in a spatial database/geographic information system (GIS), see the SAND Spatial Browser [10], [15], [66] and the QUILT system [69], [90]. Such systems find use in many application domains (e.g., digital government [38], point clouds [87] and in peer-to-peer settings [94], [95]).

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation under Grants IIS-10-18475 and IIS-12-19023, and by Google Research.

REFERENCES

[1] C.-H. Ang, H. Samet, and C. A. Shaffer. A new region expansion for quadtrees. *IEEE TPAMI*, 12(7):682–686, July 1990.

[2] W. G. Aref and H. Samet. Efficient processing of window queries in the pyramid data structure. In *PODS'90*, pp. 265–272, Nashville, TN, Apr. 1990.

[3] W. G. Aref and H. Samet. Uniquely reporting spatial objects: yet another operation for comparing spatial data structures. In *SDH'92*, pp. 178–189, Charleston, SC, Aug. 1992.

[4] W. G. Aref and H. Samet. Hashing by proximity to process duplicates in spatial databases. In *CIKM'94*, pp. 347–354, Gaithersburg, MD, Dec. 1994.

[5] D. Ayala, P. Brunet, R. Juan, and I. Navazo. Object representation by means of nonminimal division quadtrees and octrees. *TODS*, 4(1):41–59, Jan. 1985.

[6] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *SIGMOD*, pp. 322–331, Atlantic City, NJ, June 1990.

[7] F. Brabec and H. Samet. The VASCO R-tree JAVA$^{\mathrm{TM}}$ applet. In *Visual Database Systems (VDB4)*, pp. 147–153, L'Aquila, Italy, May 1998.

[8] F. Brabec and H. Samet. Visualizing and animating R-trees and spatial operations in spatial databases on the worldwide web. In *Visual Database Systems (VDB4)*, pp. 123–140, L'Aquila, Italy, May 1998.

[9] F. Brabec and H. Samet. Visualizing and animating search operations on quadtrees on the worldwide web. In *Proc. 16th European Workshop on Computational Geometry*, pp. 70–76, Eilat, Israel, Mar. 2000.

[10] F. Brabec and H. Samet. Client-based spatial browsing on the world wide web. *IEEE Internet Computing*, 11(1):52–59, Jan/Feb 2007.

[11] F. Brabec, H. Samet, and C. Yilmaz. VASCO: visualizing and animating spatial constructs and operations. In *Proc. 19th Annual Symposium on Computational Geometry*, pp. 374–375, San Diego, CA, June 2003.

[12] J.-P. Dittrich and B. Seeger. Data redundancy and duplicate detection in spatial join processing. In *ICDE*, pp. 535–546, San Diego, CA, Feb. 2000.

[13] C. R. Dyer. Computing the Euler number of an image from its quadtree. *CGIP*, 13(3):270–276, July 1980.

[14] C. R. Dyer, A. Rosenfeld, and H. Samet. Region representation: boundary codes from quadtrees. *CACM*, 23(3):171–179, Mar. 1980.

[15] C. Esperança and H. Samet. Experience with SAND/Tcl: a scripting tool for spatial databases. *JVLC*, 13(2):229–255, Apr. 2002.

[16] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *SIGMOD*, pp. 47–57, Boston, June 1984.

[17] A. Henrich. A distance-scan algorithm for spatial access structures. In *GIS'94*, pp. 136–143, Gaithersburg, MD, Dec. 1994.

[18] G. R. Hjaltason and H. Samet. Distance browsing in spatial databases. *TODS*, 24(2):265–318, June 1999.

[19] G. R. Hjaltason and H. Samet. Speeding up construction of PMR quadtree-based spatial indexes. *VLDBJ*, 11(2):109–137, Oct. 2002.

[20] G. R. Hjaltason and H. Samet. Index-driven similarity search in metric spaces. *TODS*, 28(4):517–580, Dec. 2003.

[21] E. G. Hoel and H. Samet. Efficient processing of spatial queries in line segment databases. In *SSD'91*, pp. 237–256, Zurich, Aug. 1991.

[22] E. G. Hoel and H. Samet. Benchmarking spatial join operations with spatial output. In *VLDB*, pp. 606–618, Zurich, Sept. 1995.

[23] G. M. Hunter. *Efficient computation and data structures for graphics*. PhD thesis, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.

[24] G. M. Hunter and K. Steiglitz. Operations on images using quad trees. *IEEE TPAMI*, 1(2):145–153, Apr. 1979.

[25] E. Jacox and H. Samet. Iterative spatial join. *TODS*, 28(3):268–294, Sept. 2003.

[26] E. Jacox and H. Samet. Spatial join techniques. *TODS*, 32(1):7, Mar. 2007.

[27] Y. J. Kim and J. M. Patel. Rethinking choices for multi-dimensional point indexing: making the case for the often ignored quadtree. In *CIDR 2007*, pp. 281–291, Asilomar, CA, Jan. 2007.

[28] A. Klinger. Patterns and search statistics. In J. S. Rustagi, editor, *Optimizing Methods in Statistics*, pp. 303–337. Academic Press, New York, 1971.

[29] D. E. Knuth. *The Art of Computer Programming: Sorting and Searching*, volume 3. Addison-Wesley, Reading, MA, second edition, 1998.

[30] R. Lan, M. D. Lieberman, and H. Samet. The picture of health: map-based, collaborative spatio-temporal disease tracking. In *Proc. 1st ACM SIGSPATIAL International Workshop on the Use of GIS in Public Health (HealthGIS 2012)*, Redondo Beach, CA, Nov. 2012.

[31] M. Lee, L. De Floriani, and H. Samet. Constant-time neighbor finding in hierarchical tetrahedral meshes. In *SMI'01*, pp. 286–295, Genova, Italy, May 2001.

[32] M. D. Lieberman and H. Samet. Multifaceted toponym recognition for streaming news. In *SIGIR'11)*, pp. 843–852, Beijing, July 2011.

[33] M. D. Lieberman and H. Samet. Adaptive context features for toponym resolution in streaming news. In *SIGIR'12*, pp. 731–740, Portland, OR, Aug. 2012.

[34] M. D. Lieberman and H. Samet. Supporting rapid processing and interactive map-based exploration of streaming news. In *GIS'12*, Redondo Beach, CA, Nov. 2012.

[35] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *ICDE*, pp. 201–212, Long Beach, CA, Mar. 2010.

[36] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In *GIS'07*, pp. 186–193, Seattle, WA, Nov. 2007.

[37] M. Lindenbaum, H. Samet, and G. R. Hjaltason. A probabilistic analysis of trie-based sorting of large collections of line segments in spatial databases. *SIAM J. Comp.*, 35(1):22–58, Sep. 2005.

[38] G. Marchionini, H. Samet, and L. Brandt. Introduction to the digital government special issue. *CACM*, 46(1):24–27, Jan. 2003.

[39] D. Meagher. Geometric modeling using octree encoding. *CGIP*, 19(2):129–147, June 1982.

[40] R. C. Nelson and H. Samet. A consistent hierarchical representation for vector data. *Computer Graphics*, 20(4):197–206, Aug. 1986. Also in *SIGGRAPH*, Dallas, TX, Aug. 1986.

[41] R. C. Nelson and H. Samet. A population analysis for hierarchical data structures. In *SIGMOD*, pp. 270–277, San Francisco, May 1987.

[42] S. Nutanong, E. H. Jacox, and H. Samet. An incremental Hausdorff distance calculation algorithm. *PVLDB*, 4(8):506–517, Aug. 2011.

[43] J. A. Orenstein. Multidimensional tries used for associative searching. *INFOPL*, 14(4):150–157, June 1982.

[44] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1985.

[45] G. Quercini, H. Samet, J. Sankaranarayanan, and M. D. Lieberman. Determining the spatial reader scopes of news sources using local lexicons. In *GIS'10*, pp. 43–52, San Jose, CA, Nov. 2010.

[46] J. T. Robinson. The K-D-B-tree: a search structure for large multidimensional dynamic indexes. In *SIGMOD*, pp. 10–18, Ann Arbor, MI, Apr. 1981.

[47] H. Sagan. *Space-Filling Curves*. Springer-Verlag, New York, 1994.

[48] H. Samet. Region representation: quadtrees from binary arrays. *CGIP*, 13(1):88–93, May 1980.

[49] H. Samet. Region representation: quadtrees from boundary codes. *CACM*, 23(3):163–170, Mar. 1980.

[50] H. Samet. An algorithm for converting rasters to quadtrees. *IEEE TPAMI*, 3(1):93–95, Jan. 1981.

[51] H. Samet. Computing perimeters of images represented by quadtrees. *IEEE TPAMI*, 3(6):683–687, Nov. 1981.

[52] H. Samet. Connected component labeling using quadtrees. *JACM*, 28(3):487–501, July 1981.

[53] H. Samet. Distance transform for images represented by quadtrees. *IEEE TPAMI*, 4(3):298–303, May 1982.

[54] H. Samet. Neighbor finding techniques for images represented by quadtrees. *CGIP*, 18(1):37–57, Jan. 1982.

[55] H. Samet. A quadtree medial axis transform. *CACM*, 26(9):680–693, Sept. 1983. Also see CORRIGENDUM, *CACM*, 27(2):151, Feb. 1984.

[56] H. Samet. Algorithms for the conversion of quadtrees to rasters. *CVGIP*, 26(1):1–16, Apr. 1984.

[57] H. Samet. Reconstruction of quadtrees from quadtree medial axis transforms. *CVGIP*, 29(3):311–328, Mar. 1985.

[58] H. Samet. A top-down quadtree traversal algorithm. *IEEE TPAMI*, 7(1):94–98, Jan. 1985.

[59] H. Samet. An overview of quadtrees, octrees, and related hierarchical data structures. In R. A. Earnshaw, editor, *Theoretical Foundations of Computer Graphics and CAD*, pp. 51–68. Springer-Verlag, Berlin, West Germany, 1988.

[60] H. Samet. Implementing ray tracing with octrees and neighbor finding. *Computers & Graphics*, 13(4):445–460, 1989.

[61] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley, Reading, MA, 1990.

[62] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.

[63] H. Samet. Decoupling partitioning and grouping: overcoming shortcomings of spatial indexing with bucketing. *TODS*, 29(4):789–830, Dec. 2004.

[64] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan-Kaufmann, San Francisco, 2006.

[65] H. Samet. K-nearest neighbor finding using MaxNearestDist. *IEEE TPAMI*, 30(2):243–252, Feb. 2008.

[66] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. *CACM*, 46(1):63–66, Jan. 2003.

[67] H. Samet, M. D. Adelfio, B. C. Fruin, M. D. Lieberman, and B. E. Teitler. Porting a web-based mapping application to a smartphone app. In *GIS'11*, pp. 525–528, Chicago, November 2011.

[68] H. Samet, B. C. Fruin, and S. Nutanong. Duking it out at the smartphone mobile app mapping API corral: Apple, Google, and the competition. In *Proc. 1st ACM SIGSPATIAL Int. Wrkshp on Mobile Geog. Inf. Sys. (MobiGIS 2012)*, Redondo Beach, CA, Nov. 2012.

[69] H. Samet, A. Rosenfeld, C. A. Shaffer, and R. E. Webber. A geographic information system using quadtrees. *Pattern Recognition*, 17(6):647–656, Nov/Dec 1984.

[70] H. Samet, J. Sankaranarayanan, and H. Alborzi. Scalable network distance browsing in spatial databases. In *SIGMOD*, pp. 43–54, Vancouver, Canada, June 2008.

[71] H. Samet and C. A. Shaffer. A model for the analysis of neighbor finding in pointer-based quadtrees. *IEEE TPAMI*, 7(6):717–720, Nov. 1985.

[72] H. Samet, C. A. Shaffer, and R. E. Webber. Digitizing the plane with cells of non-uniform size. *INFOPL*, 24(6):369–375, Apr. 1987.

[73] H. Samet and M. Tamminen. Bintrees, CSG trees, and time. *Computer Graphics*, 19(3):121–130, July 1985. Also in *SIGGRAPH*, San Francisco, July 1985.

[74] H. Samet and M. Tamminen. Computing geometric properties of images represented by linear quadtrees. *IEEE TPAMI*, 7(2):229–240, Mar. 1985.

[75] H. Samet and M. Tamminen. An improved approach to connected component labeling of images. In *CVPR*, pp. 312–318, Miami Beach, FL, June 1986.

[76] H. Samet and M. Tamminen. Efficient component labeling of images of arbitrary dimension represented by linear bintrees. *IEEE TPAMI*, 10(4):579–586, July 1988.

[77] H. Samet, B. E. Teitler, M. D. Adelfio, and M. D. Lieberman. Adapting a map query interface for a gesturing touch screen interface. In *WWW'11 (Companion Volume)*, pp. 257–260, Hyderabad, India, Mar. 2011.

[78] H. Samet and R. E. Webber. On encoding boundaries with quadtrees. *IEEE TPAMI*, 6(3):365–369, May 1984.

[79] H. Samet and R. E. Webber. Storing a collection of polygons using quadtrees. *TOGS*, 4(3):182–222, July 1985.

[80] J. Sankaranarayanan, H. Alborzi, and H. Samet. Efficient query processing on spatial networks. In *GIS'05*, pp. 200–209, Bremen, Germany, Nov. 2005.

[81] J. Sankaranarayanan, H. Alborzi, and H. Samet. Distance join queries on spatial networks. In *GIS'06*, pp. 211–218, Arlington, VA, Nov. 2006.

[82] J. Sankaranarayanan and H. Samet. Distance oracles for spatial networks. In *ICDE*, pp. 652–663, Shanghai, Apr. 2009.

[83] J. Sankaranarayanan and H. Samet. Query processing using distance oracles for spatial networks. *IEEE TKDE*, 22(8):1158–1175, Aug. 2010.

[84] J. Sankaranarayanan and H. Samet. Roads belong in databases. *IEEE Data Engineering Bulletin*, 33(2):4–11, June 2010.

[85] J. Sankaranarayanan, H. Samet, and H. Alborzi. Path oracles for spatial networks. *PVLDB*, 2(1):1210–1221, Aug. 2009.

[86] J. Sankaranarayanan, H. Samet, B. Teitler, M. D. Lieberman, and J. Sperling. TwitterStand: News in tweets. In *GIS'09*, pp. 42–51, Seattle, WA, Nov. 2009.

[87] J. Sankaranarayanan, H. Samet, and A. Varshney. A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2):157–174, Apr. 2007.

[88] T. Sellis, N. Roussopoulos, and C. Faloutsos. The R^+-tree: a dynamic index for multi-dimensional objects. In *VLDB*, pp. 71–79, Brighton, United Kingdom, Sept. 1987.

[89] C. A. Shaffer and H. Samet. Optimal quadtree construction algorithms. *CVGIP*, 37(3):402–419, Mar. 1987.

[90] C. A. Shaffer, H. Samet, and R. C. Nelson. QUILT: a geographic information system based on quadtrees. *IJGIS*, 4(2):103–131, Apr.–June 1990.

[91] R. Sivan and H. Samet. Algorithms for constructing quadtree surface maps. In *SDH'92*, volume 1, pp. 361–370, Charleston, SC, Aug. 1992.

[92] M. Tamminen and H. Samet. Efficient octree conversion by connectivity labeling. *Computer Graphics*, 18(3):43–51, July 1984. Also in *SIGGRAPH*, Minneapolis, MN, July 1984.

[93] S. L. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *CGIP*, 4(2):104–119, June 1975.

[94] E. Tanin, A. Harwood, and H. Samet. A distributed quadtree index for peer-to-peer settings. In *ICDE*, pp. 254–255, Tokyo, Apr. 2005.

[95] E. Tanin, A. Harwood, and H. Samet. Using a distributed quadtree index in P2P networks. *VLDBJ*, 16(2):165–178, Apr. 2007.

[96] B. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: A new view on news. In *GIS'08*, pp. 144–153, Irvine, CA, Nov. 2008.

[97] W. Wang, J. Yang, and R. Muntz. PK-tree: a spatial index structure for high dimensional point data. In *FODO'90*, pp. 27–36, Kobe, Japan, Nov. 1998.