DATA STRUCTURES FOR COMPUTER GRAPHICS

L. De Floriani - G. Dettori
Istituto per la Matematica Applicata del C.N.R.
Via L.B. Alberti,4 - 16132 Genova (Italy)

Keywords
Computer Graphics, Image Processing, Computer Aided Design, Data Structures,
Lists, Trees, Graphs, Hierarchies, Relations.

ABSTRACT

The use of convenient data structures which allow a fast access to the data is a crucial problem in computer graphics to produce efficient representation and manipulation algorithms. The choice of a data structure depends on the problem under consideration, i.e. on the operations to be performed on the data. Thus, different kinds of data structures have been developed for graphical applications, both as working and as representation structures. In this paper we describe and analyse the features of some well-known hierarchical and relational structures which are widely employed in computer graphics.

1. INTRODUCTION

Computer Graphics deals with pictures of different kinds, since display devices available for this purpose, like CRT, plotters, raster scans, etc., show a variety of characteristics. In many cases, the large amount of information involved makes picture generation a complex and rather inefficient task. Hence, the choice of convenient data structures is a crucial problem, which strongly affects the performance of graphical algorithms. A fast access and manipulation of the data is a fundamental requirement, especially in real-time applications /29,36,39,48/.

The generation of complex pictures may require that different data structures are employed during the development of this process. This leads to a first classification of data structures into representation and working structures. The former ones model the features of an object according to the problem under consideration and to the operations to be performed on it, whereas the latter are chosen in order to fit the characteristics of the display surfaces, i.e. based on sets of pixels or on sets of vectors.

Arrays in one, two or three dimensions have been the first and most common data structure employed in graphical algorithms, either because they are the only structure allowed by programming languages in use during the first decade of computer graphics, and also since arrays seem to be the most natural way to encode images as collections of pixels and contours as sequences of points.

A major gain in efficiency is obtained by using dynamical structures, like linked lists, trees or graphs, provided by more recent high-level programming languages, which allow a fast access to and updating of information.

A linear list, defined as a finite linearly ordered set of items /23/, is the simplest dynamical data structure and is suitable for handling data to be processed sequentially. For instance, sorted lists are used for
describing polygonal lines as chains of vertices and edges, while stacks and queues are mostly employed as internal storage structures /36/.

Trees and graphs are structures of major interest, since their flexibility has led to several variants, according to different representation requirements. Both of them can be defined as a pair \( T = (N, R) \), where \( N \) is a finite set of elements, called nodes, and \( R \) is a binary relation, which establishes a sorting on the set of nodes.

The properties of the relation determine the characteristics of the structure and distinguish between the two. In a tree, each node has one ancestor or parent and several descendants, or children. A single node, called root, has no ancestor, as well as a set of nodes, called leaves, have no descendants. On the other hand, in a graph each node may have an arbitrary number of ancestors and descendants. Hence, trees correspond to the concept of hierarchy, since they can provide representations of objects at successive levels of increasing resolution. On their side, graphs are suitable to describe the mutual relationships between the various parts of a picture or of a graphical object.

The purpose of this paper is to describe and analyze the hierarchical and relational structures used in Computer Graphics. We analyze the features of some well-known hierarchical structures, namely segmentation trees, concavity trees, striptrees and extentions, quadtrees and octrees. As regards relational structures, we take into consideration winged graphs, triangulations and adjacency graphs. The applications suggested for both structures are taken in the field of Computer Graphics. It must be pointed out, however, that most of the data structures considered here find application also in Image Processing and Pattern Recognition, since both disciplines manipulate pictorial data /39/.

2. HIERARCHICAL DATA STRUCTURES

Among the existing dynamical information structures, trees are those which better correspond to the concept of hierarchy.

In this regard, it can turn out useful to recursively define a non-empty tree as a non-empty, finite set of labelled nodes such that there is one distinguished node, called root, and the remaining nodes are partitioned into a number of disjoint subtrees. The number of subtrees obtained by deleting the root, if constant, determines the \textit{arity} of the tree.

Since the representations employed in computer graphics can be either attached to a fixed grid or to the considered objects themselves, there are two basic approaches to the hierarchical description of pictures: domain-dependent or object-dependent. Segmentation, concavity and striptrees, which are based on different features of the shape, belong to the latter class, while quadtrees, which describe both contours and images in terms of the pixels of a superimposed grid, belong to the former one.

2.1 Segmentation trees

Trees with nodes having a variable number of children are used in image segmentation both as working and as representation structures. An example of their application as working structures is provided by the subdivision of a picture into homogeneous (for instance, monochromatic) subpictures. This segmentation problem can be stated as follows /22/.

Let \( X \) be a picture domain, \( S \) a subdomain of \( X \) and \( P(S) \) a boolean predicate which is \textit{true} if \( S \) is monochromatic, \textit{false} otherwise. A segmentation of \( X \) consists of a partition of \( X \) into subdomains \( S_i \), \( i = 1, 2, \ldots, n \), such that

\[
\begin{align*}
(1) & \quad X = \bigcup_{i=1}^{n} S_i \\
(2) & \quad S_j \cap S_k = \emptyset \text{ for every } i \neq j \\
(3) & \quad P(S_i) = \text{true} \text{ for every } i \\
(4) & \quad P(S_j \cap S_k) = \text{false} \text{ for every } i \neq j
\end{align*}
\]
Horowitz and Pavlidis /22/ suggest a tree structure suitable for picture segmentation. The root of the tree corresponds to the whole picture domain \( \mathcal{X} \), the children of a node represent a partition of the region associated with their parent. Fig. 1 shows an example of picture segmentation and the corresponding tree. This structure is the basis for a split-and-merge algorithm for region segmentation, which alternatively splits a region into subregions and then merges adjacent subregions having the same color.

![Figure 1](image1.png)

**Fig. 1** An example of region segmentation and corresponding tree structure.

### 2.2 Concavity Trees

A graphical application of general trees as a representation structure is given by concavity trees /3,6,50/. This structure, as suggested by its name, allows the description of polygons in terms of their convex hulls and concavities. Given a polygon of arbitrary shape, such a tree contains the convex hull of the polygon as root and every other node represents the convex hull of a hole in the polygon associated with its parent (see Fig. 2). Advantages of this representation include invariance with respect to rotation and possibility of reflection. Its drawback is its instability, since any slight change in the input strongly affects the tree definition.

![Figure 2](image2.png)

**Fig. 2** A polygon and its concavity tree

Concavity trees can be employed in a similar way for representing polyhedra. In this case, the root of the tree contains the three-dimensional convex hull of the polyhedron, and any other node the convex hull of a hole. This structure shows similar characteristics to those of the two-dimensional one. It is, however, not widely used because it requires a great computational effort, which makes other structures preferable for solid object representation /5/.
2.3 Striptrees and extensions

Striptrees represent a well-known application of binary trees as a representation structure /4/. They are used for representing planar curves at different levels of resolution. The root of the tree corresponds to the minimum strip, containing the entire curve, parallel to the direction of the segment joining its extreme points. The children of a node are computed by subdividing into two pieces the part of the curve corresponding to this node. The subdivision is performed at that point at which the maximum distance from the approximating segment is achieved (see fig. 3). The resolution in the representation is given by the maximum width of the strips corresponding to the leaves. Representing curves by means of striptrees allows a simple and efficient computation of their length, union and intersection.

Fig. 3 A striptree representation of a planar curve

In general, curves can be represented by binary segmentation trees obtained by recursively splitting them in two parts, until a boolean predicate defined according to some application requirements assumes the value false /28/. This approach can be generalized to represent 2-1/2D surfaces by networks of nested non-overlapping triangular patches /13/. By this technique a ternary tree is generated, in which the root represents the initial triangle and every other node a triangle obtained from the subdivision of its parent triangle into three subtriangles.

Such a structure produces a variable resolution representation of a surface defined as a set of randomly located points (see fig. 4).

Fig. 4 An example of hierarchical triangulation and corresponding tree structure.

For the representation of both surfaces and spatial curves, prismtrees have been developed, which constitute a generalization of striptrees obtained by replacing rectangular strips by prisms /15/.

2.4 Quadtrees and Octrees

The quadtree is the most suitable hierarchical structure for describing raster figures. It is widely employed in graphical representation /6,24,28,54/, because it gives a compact but exhaustive description of an image and also produces approximations of a picture at varying resolutions. Quadtrees are defined according
to the principle of recursive decomposition and are mostly used for handling raster images composed by 2 x 2 pixels taking the value 0 or 1. A square picture is subdivided into four square subpictures which are in turn subdivided until homogeneous subregions are found. This decomposition process leads to a tree whose non-terminal nodes have four children (see fig. 5). Leaves are classified as WHITE or BLACK depending on whether they correspond to blocks composed only by 0's or 1's. Non-terminal nodes are called GRAY.

Quadtrees can be employed for representing different kinds of images. Various algorithms have been developed for converting the most common representations such as raster, boundary code /18/, or binary array into quadtrees, and conversely /14,41-44/. Using quadtrees permits to efficiently perform union, intersection, difference, and clipping of images /25/ and to compute geometric properties of a region, such as area, perimeter, moments, connected components /45,46,49/. Thus, their main application fields include raster graphics, image analysis and pattern recognition. Algorithms for linear transformations of images represented as quadtrees have also been developed /24-26/. Sanet /46/ gives a complete survey of quadtrees and similar hierarchical data structures used for representing curves and point data. In picture generation the quadtree has been used when deriving a graphical representation from volume models, since in this case there is the need of maintaining and updating a synthesized image when evaluating the model /54/.

Quadtrees are also used as a working structure, as, for instance, in Warnock's algorithm for hidden surface elimination /51/. The novelty of this technique was that it implements a divide-and-conquer approach on the picture area and not on the scene to be processed. The picture area is subdivided until either an area is detected which is sufficiently simple to be displayed, or the resolution of the device is reached.

Binary trees are presently studied as an alternative structure to quadtrees for image representation. They seem to allow the development of more efficient algorithms since they require less storage space as searching complexity /39/.

The 3-D generalization of quadtrees, called octrees, has been widely studied and is actually used for representing solid objects /27, 34/. They are based on a recursive subdivision of the object domain into octants. Leaves correspond to homogeneous cubes completely internal or external to the object (see fig. 6). The use of octrees allows an easy representation of 3-D objects on raster displays in the form of a quadtree.
which can be straightforwardly constructed from the given structure. Like quadtrees, octrees are well suited for boolean and geometrical operations. They need, however, a large amount of storage space when the space of the object under consideration requires a high resolution representation.

3. RELATIONAL DATA STRUCTURES

A different description of an image can be defined when regarding it as a set of interrelated primitive components. In this case, relational data structures are used as internal representation of the form of the object. When one or more binary relations can be defined on the basic components of an object, then graphs can be successfully employed as representation or working structures.

Graphs, and more generally relational structures, are extensively used for object representation in almost every application field concerning pictorial output and/or input. In this section we focus the attention on three graph structures commonly used in computer graphics: Winged Graphs, employed as models of the boundary of polyhedral objects /5,7/, Triangulations, used as the basis of surface models /47/, and Adjacency Graphs, which represent the adjacency relations between parts of a raster image /28/.

3.1 Winged Graphs

An exhaustive description of the boundary of two- or three-dimensional objects is given by the so called Winged Graphs /7,11/. In two dimensions, the boundary of a region can be simply represented as a circular linked list. A more complete representation, however, can be obtained by explicitly storing the fundamental entities which form the boundary of a region, i.e. contours, edges and vertices, and their mutual relationships. As the example of fig. 7 shows, the relations between the set of contours bounding a region are modeled by a pointer structure in which downward links represent contours lying inside the area of the corresponding contour.
Each contour is in turn represented as a doubly linked chain of edge nodes, each of which is also connected to the vertex nodes corresponding to its endpoints (see fig. 8). This graph-based structure has been proved to be a powerful tool for polygon comparison operations, i.e. for solving polygon set and clipping problems by a single process /53/. Because of the kind of topological information explicitly stored, this structure can be useful to examine adjacencies between contours when performing operations such as antialiasing on an image in a frame buffer or manipulation of hidden surfaces or hue scenes.

![Diagram of winged graph representation of contour C3 of fig. 7.](image)

In three dimensions, winged graphs seem to be the most suitable and efficient data structure for representing the boundary of polyhedral objects. A boundary model of an object consists of a segmentation of its boundary into a finite number of boundary subsets, called faces, which are in turn represented as the collection of their bounding edges and vertices. Such a representation is particularly efficient in computer graphics, for visible surface display, while volumetric models, which describe an object in terms of their primitive volume elements, are considered better suited to object manipulation and recognition for applications in Computer Aided Design and Computer Vision. In a winged graph, there are four kinds of nodes, corresponding respectively to the vertices, edges, faces and shells of the corresponding object. These nodes are arranged in circular doubly linked lists. The shell nodes are organized in the same way as the contour nodes in the two-dimensional example. Each face node contains a pointer to one of its boundary edges, and each vertex node points to one of the edges incident in it. Each edge node contains links to the two faces sharing it and to its end points. Such pointers represent the two binary adjacency relationships defined over the set of faces and vertices of an object by its edges. This data structure allows a fast access to information related to vertices, edges and faces, as well as a computational efficiency in performing operations useful both in model construction and in hidden line or surface picture generation.

3.2 Triangulations

A simpler graph structure is used to encode the boundary of objects when their surface is partitioned into a set of non-overlapping triangle patches. A surface decomposition into triangles is represented as a graph \( T \), called triangulation, having vertices at a set of points \( S \) on the surface, edges joining pairs of points of \( S \), in such a way that every face of \( T \) is a triangle /12/.

The case of 2+1/2 dimensional surfaces has been more extensively studied, especially for Digital Terrain Model construction /17,31/. In this case the problem of approximating a surface defined at a set of arbitrarily spaced data reduces to that of computing a triangulation of the surface domain with vertices at the data points projections. Although there exists many possible triangulations of a set of points in the plane, that to be used as domain discretization for surface approximation must satisfy an optimality criterion based on the maximum of the minimum angle of the triangles forming it /31,47/. An optimal triangulation in this sense is called Delaunay triangulation (see fig. 9). Efficient algorithms which compute it in \( O(N) \) or \( O(N \log N) \) time have been recently exhibited in the literature /12,20,32,33,47/.
Fig. 9 Delaunay triangulation of a set of points in the plane

The problem of triangulating the boundary of a closed surface has been studied for 3D object modeling in CAD, Computer Vision and Robotics /9,16/. One common approach, which works only for convex surfaces, consists of projecting on the plane the points defining the surface and triangulating the resulting set. Some recently proposed algorithms construct a triangulation of a surface in the case this is defined as a set of parallel contours /19/. Other schemes compute the 3D Delaunay tessellation of the convex hull of the points defining the surface, then eliminate tetrahedra until all the vertices of the tessellations lie on the boundary /10/.

A triangulation can be internally encoded either as the adjacency list of its vertices and edges or in the form of a list of triangles and their mutual adjacency relations /12/. This latter data structure allows an efficient encoding of the shape of the triangular patches, since geometrical information can be straightforwardly attached to each triangle.

3.3 Adjacency Graphs

Adjacency Graphs are employed in Computer Graphics as an internal representation structure of a picture segmented into a set of simple components, each corresponding to one of its parts. In raster graphics, pixel-based techniques for contour filling are examples of algorithms which make use of a traversal strategy applied to the Line Adjacency Graph (LAG) of the picture /38/.

A LAG corresponds to a subdivision of a picture into strips parallel to the horizontal coordinate axis. A different node of the LAG is associated with each interval in a strip which belongs to the object. Every pair of nodes belonging to adjacent strips and corresponding to contiguous parts of the object are then connected by an edge (see fig. 10). The graph can be sometimes considered directed because the nodes can be ordered according to their vertical coordinate. A LAG is used for image segmentation and in pattern recognition /39/, because it is better suited than a linear list to exploit the two-dimensional structure of an image.

Fig. 10 An image and its line adjacency graph (LAG)
When the representation of the connections among arbitrarily shaped components of a picture is required, a Region Adjacency Graph (RAG) should be used /52/. In a RAG nodes represent subpictures and edges adjacency relations between them. In fact, a pair of nodes is connected by an edge if and only if the regions associated with them share a common side (see fig. 11). Thus, the degree of a node is equal to the number of regions which are adjacent to the one associated with that node. A cutnode in the graph /21/ corresponds to a region which completely encloses some other ones. Therefore, this and other topological information about the picture can be obtained by simply checking the topological properties of the RAG /52/.

4. CONCLUDING REMARKS

The choice of a suitable data structure is a crucial problem in the design of algorithms for representation and manipulation of pictures. Structures of different kind have to be used according to the requirements of the problem under consideration.

A first criterion to rely on for choosing a suitable structure is whether the representation is raster or vector. For raster pictures, quadtreess and their alternative binary trees are commonly used both as working and as representation structures. Besides the quadtree, the LAG is convenient in those problems, like contour filling, where objects are processed line by line. Representations based on space enumeration, in particular octrees, are suitable for encoding 3D objects to be displayed on a raster scan.

As concerns vector graphics, lists or stritrees are to be chosen for representing contours, while concavity trees, winged graphs and RAG's describe planar objects as two dimensional entities. For three dimensional objects, the choice is among winged graphs and triangulations, hierarchical triangulations, prism trees and octrees. Winged graphs and triangulations are employed to represent the boundary of an object; the former are required when the boundary is decomposed into arbitrarily shaped faces; the latter when triangular patches only are considered. The hierarchical triangulation combines the advantages of triangular grids with those deriving from the use of a variable resolution surface model. Prisntrees allow a representation of both curves and boundaries of objects at different resolutions. Finally, octrees describe objects as volumes at increasing levels of accuracy.

In general, when selecting a data structure, it must be kept in mind that hierarchical structures implement the concept of increasing resolution, as well as relational ones correspond to the idea of

![Fig. 11: An image and its region adjacency graph (RAG)](image-url)
segmentation. Hierarchical graphs can also be constructed, which represent a suitable combination of both concepts /2,5/.

ACKNOWLEDGEMENTS

The authors are grateful to Sandra Burlando for her help in typing this paper.

REFERENCES


/10/ J.D. Boissonnat, "Representing 2-D and 3-D Shapes with the Delaunay Triangulation", Proceedings ICPR '84, Montreal, 1984, pp. 745-748.


/21/ P. Harary, Graph Theory, Addison Wesley, 1969.


