GIOTTO3D: A System for Visualizing Hierarchical Structures in 3D*

Ashim Garg and Roberto Tamassia

Department of Computer Science
Brown University
Providence, RI 02912–1910, USA
{ag,rt}@cs.brown.edu

Abstract. Hierarchical structures represented by directed acyclic graphs are widely used in visualization applications (e.g., class inheritance diagrams and scheduling diagrams). 3D information visualization has received increasing attention in the last few years, motivated by the advances in hardware and software technology for 3D computer graphics. We present GIOTTO3D, a system for visualizing hierarchical structures in 3D. GIOTTO3D uses a new technique combining 2D drawing methods with a lifting transformation that exploits the third dimension to visualize hierarchical relations among the vertices. GIOTTO3D also employs several graphical aids such as user-defined coloring, showing/hiding sub-hierarchies, "footprints", and representation of edges as "Bezier tubes" to improve the effectiveness of its visualizations.

1 Introduction

The effective visualization of hierarchical structures is an important problem in the area of advanced visual interfaces. Hierarchical structures occur in a wide variety of information visualization applications, including WWW-navigation, business graphics, multimedia documents, software engineering, algorithm animation, planning, database design, and visual languages.

A hierarchical structure can be formally modeled by a *directed acyclic graph* (*DAG*), i.e., a graph with directed edges and no directed cycles. Hence, throughout this paper, we will use the two terms interchangeably. Also we will use the terms visualization, drawing, layout and representation to denote the same concept. A hierarchical structure is usually visualized with a *hierarchical drawing* where each edge is drawn as a curve monotonically non-increasing in the y-direction for 2D drawings and in the z-direction for 3D drawings.

Three dimensions offer many advantages over two dimensions for visualizing hierarchical structures. The extra dimension gives greater flexibility for placing the vertices and edges of the graph. The presence of navigational operations such as rotation, translation, and zooming not only helps the user in understanding the structure, but also results in a more effective use of screen space because not every object has to be shown in front on the screen (as is the case with 2D-visualization). Empirical studies have also shown that 3D visualizations are generally easier to understand than 2D visualizations [21, 26]. Many

*Research supported in part by the National Science Foundation under grant CCR–9423847, and by the U.S. Army Research Office under grant DAAH04–96–1–0013.

virtual-reality packages also accept descriptions of 3D layouts produced by other graphics packages.

Research in graph drawing has generally concentrated on constructing two-dimensional visualizations because of the two-dimensional geometry of display surfaces such as paper and computer screen. Recent advances in hardware and software technology for computer graphics open up the possibility of displaying three-dimensional (3D) visualizations on a variety of low-cost workstations, and several researchers (and film makers**) have begun to explore the possibility of displaying graphs and networks using this new technology. Indeed, it is expected that in the next few years, 3D visualization will become widespread in Web documents, thanks to the VRML language for modeling three-dimensional scene graphs. Recent work on WWW navigation [18], software visualization [19], information visualization [20], and algorithm animation [1] has also advocated the use of 3D visual representations.

Previous research on 3D graph drawing (both hierarchical and non-hierarchical) has focused on the development of visualization systems (see, e.g., [2, 5, 17]). Recent theoretical work has been reported in [3, 4, 15].

We now review previous work on visualizing hierarchical structures in 3D. In a *cone tree* [20], each subtree is associated with a cone such that the vertex at the root of the subtree is placed at the apex of the cone and its children are circularly arranged around the base of the cone. SemNet [10] is a system for displaying knowledge bases. GraphVizualizer3D [27] emphasizes manual layouts. COMAIDE [9] uses a force-directed method. The systems GMB [16] and PLUM [19] extend to 3D the layering approach [23, 13] which was conceived for constructing 2D hierarchical drawings. The drawings constructed by these two systems have characteristics similar to their 2D counterparts.

2 GIOTTO3D

GIOTTO3D is a system for visualizing hierarchical structures in 3D. Namely, given a DAG G as input, it constructs a 3D hierarchical drawing of G.

In Section 2.1, we describe the 3D drawing algorithm used by GIOTTO3D and discus graphical aspects. In Section 2.2, we give examples of drawings produced by and applications of GIOTTO3D.

2.1 Drawing Algorithm

GIOTTO3D uses a novel combination of 2D undirected layout methods, which guarantee the satisfaction of several aesthetic criteria, and a lifting transformation that uses the third dimension to visualize the hierarchical relationships between the vertices.

There is considerable work on 2D drawings of undirected graphs [6], and many algorithms have been developed for constructing drawings satisfying one or more aesthetic criteria.

** An important plot element in the movie *Jurassic Park* involves a 3D virtual-reality traversal of a tree representing a Unix file system.

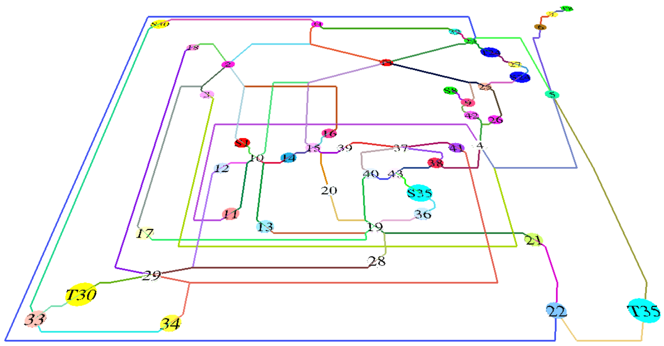


Fig. 1. 2D polyline drawing constructed by Phase *Draw-Flat* of GIOTTO3D.

GIOTTO [25] is a successful general-purpose drawing algorithm based on the *planarization* approach and a bend-minimization method. Namely, given a non-planar graph as input, it first transforms the graph into a planar graph by replacing each crossing with a fictitious vertex, and then constructs a 2D orthogonal drawing with the bend-minimization method of [24]. GIOTTO produces high-quality layouts and has been widely used in software visualization systems. In [7, 8], the performance of GIOTTO on more than 11,000 graphs derived from "real-life" software engineering and database applications was experimentally evaluated. This study show that GIOTTO performs very well with respect to several quality measures such as crossings, bends, total edge-length and area.

The main idea behind the approach of GIOTTO3D is to leverage the aesthetic quality of the 2D layouts produced by GIOTTO. Given a DAG G as input, GIOTTO3D constructs a 3D drawing of G in three phases:

1. *Draw-Flat:* Construct a 2D polyline drawing of G in the XY-plane using a variation of GIOTTO.
2. *Lift:* Assign z-coordinates to the vertices and to the bends of the edges such that their placement reflects the hierarchy.
3. *Reshape:* Draw vertices as spheres and edges as Bezier tubes [22], and create a footprint of the 3D drawing.

We now describe each step in more detail.

Draw-Flat. In this phase, GIOTTO3D constructs a 2D polyline drawing of G in the XY-plane using a variation GIOTTO. Figure 1 shows an example 2D drawing constructed in Step *Draw-Flat* by GIOTTO3D.

GIOTTO works in two steps:. In the planarization step, G is converted into a planar graph G' by replacing crossings with fictitious vertices. In the orthogonalization step, an orthogonal drawing D of G' is constructed using the bend-minimization algorithm of [24]. In the drawing D produced by GIOTTO, the vertices are drawn as rectangles with possibly different sizes. Thus, we have added a postprocessing step that shrinks all the vertices to the same size. Additional segments are added to the edges to maintain connectivity, whenever needed. The output of this phase is a 2D polyline drawing of G.

Lift. In this phase, we assign z-coordinates to the vertices, and to the bends of the edges such that each edge is drawn as a polygonal chain nonincreasing in the z-direction. We use the following approach for reducing the total length of the edges, which is common to many layering based methods (see, e.g., [13]). We call a *source* of G a vertex without incoming edges. The z-coordinates of the vertices are given by an optimal solution to the following integer linear program:

$$\min(\textstyle\sum_{v \to w} z(v) - z(w))$$
subject to: $z(v) \geq z(w) + 1$ for each edge $v \to w$ of G, and
$z(u) = 0$ for each source vertex of G.

Although solving an integer linear program is NP-hard in general, we have implemented a fast heuristic based on topological numbering for finding a good solution.

Reshape GIOTTO3D draws vertices as spheres. We chose this representation because of the good visual properties of spheres, such as reflectivity, roundness, smoothness, and symmetry. Also, after the postprocessing step of the *Draw-Flat* phase, the end-segments of each edge are directed towards the centers of their incident vertices. Thus, by drawing vertices in 3D as spheres, it is likely that their incident edges will enter along a direction normal to the surface of the sphere. This feature adds to the visual appeal of the drawing. The vertices can be assigned colors by an application. This allows the users to encode application-specific information with colors.

Each edge is represented by a *Bezier tube* [22] whose control points are given by the bend-points of the edge. A Bezier tube is a Bezier surface [11] that is shaped like a hollow tube. Bezier tubes were introduced in [22], and were shown to have good visual properties, such as reflectivity, roundness, and smoothness. We have also experimented with drawing edges as Bezier curves and as a set of connected rigid pipes. Bezier tubes give by far the best results. Other than [22], we do not know of any other visualization system that uses Bezier tubes. Edges are given randomly generated colors over a wide range. This reduces the possibility that two edges have the same color, and helps in distinguishing them if they are drawn close to each other.

Previous approaches have recommended displaying idealized shadows of the objects of a 3D structure to assist in understanding it [20]. GIOTTO3D follows a different approach. Instead of displaying idealized shadows, GIOTTO3D displays the *footprints* of the graphical objects. A footprint of an object is its projection on the XY-plane(see Fig. 2(a)). The footprint of a displayed structure is similar to its 2D-drawing constructed by Step *Draw-Flat*.

The footprint displays the hierarchy-independent connectivity information of the structure, and has few crossings. The use of the same coloring scheme for both 3D visualization and footprint helps in maintaining the correspondence between them.

GIOTTO3D provides the usual 3D navigational operations: rotation, translation, and zooming. Fog and appropriate lighting can be used to enhance depth perception. A user can also select a subgraph and perform various operations on it, including showing (hiding) the selected subgraph, or showing (hiding) the descendants of selected vertices. Users can also choose to either show an outline of the hidden objects or not show them at all.

2.2 Example Layouts and Applications

Hierarchical structures are common in real-life applications. In this section, we illustrate the use of GIOTTO3D in three applications. Sample GIOTTO3D visualizations are also shown (with color images) in [14].

Education. Figure 2 shows two different views (from different camera angles and at different zooms) of a DAG depicting the evolution-history of UNIX, taken from [13].

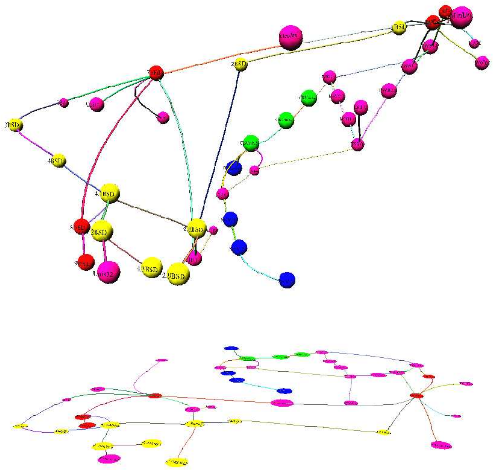
Economic Planning. Figure 3(a) shows a drawing of a subgraph of the dynamic model of world interactions given in [12], which shows the interdependency of five important economic variables — population, capital investments, natural resources, fraction of capital devoted to agriculture, and pollution.

Data Structures. Figure 3(b) shows a complete binary tree with levels. While a complete binary tree can be effectively visualized in 2D, a 3D representation uses screen space more efficiently than a 2D representation [20].

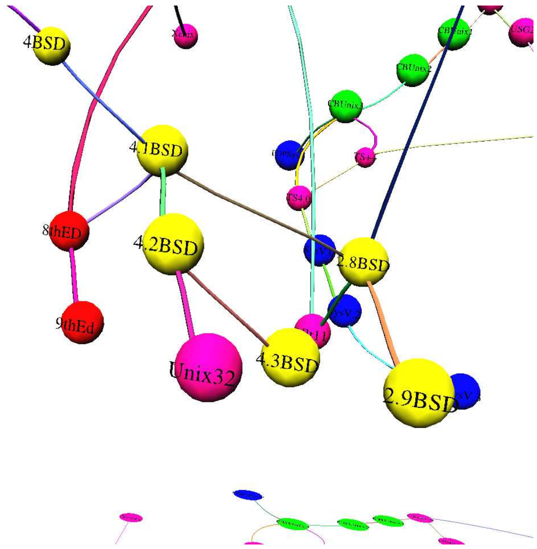
References

1. M. Brown and M. Najork. Algorithm animation using 3D interactive graphics. In *ACM Symp. on User Interface Software and Tech.*, pages 93–100, 1993.
2. I. Bruß and A. Frick. Fast interactive 3-d graph visualization. *Graph Drawing (Proc. GD '95). LNCS*, 1027:99–110, Springer-Verlag. 1996.
3. M. Chrobak, M. T. Goodrich, and R. Tamassia. Convex drawings of graphs in two and three dimensions. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 319–328, 1996.
4. R. F. Cohen, P. Eades, T. Lin, and F. Ruskey. Three-dimensional graph drawing. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes in Computer Science*, pages 1–11. Springer-Verlag, 1995.
5. I. F. Cruz and J. P. Twarog. 3d graph drawing with simulated annealing. *Graph Drawing (Proc. GD '95). LNCS*, 1027:162–165, Springer-Verlag. 1996.
6. G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Comput. Geom. Theory Appl.*, 4:235–282, 1994.

7. G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimental comparison of three graph drawing algorithms. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 306–315, 1995.
8. G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimental comparison of four graph drawing algorithms. *Comput. Geom. Theory Appl.*, 1996. to appear.
9. D. Dodson. Comaide: Information visualization using cooperative 3d diagram layout. *Graph Drawing (Proc. GD '95). LNCS*, 1027:190–201, Springer-Verlag. 1996.
10. K. M. Fairchild, S. E. Poltrock, and G. W. Furnas. Semnet: Three-dimensional graphic representation of large knowledge bas es. In *Cognitive Sc. and its Applns for Human-Computer Interaction*, pages 201–233. Lawrence Erlbaum Assoc., 1988.
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. J. W. Forrester. *World Dynamics*. Wright-Allen, Cambridge, Mass., 1971.
13. E. R. Gansner, E. Koutsofios, S. C. North, and K. P. Vo. A technique for drawing directed graphs. *IEEE Trans. Softw. Eng.*, 19:214–230, 1993.
14. A. Garg and R. Tamassia. Effective visualization of hierarchical structures in 3D. Manuscript, Dept. of Computer Sci., Brown University, 1996. Available at *http://www.cs.brown.edu/people/rt/fadiva/giotto3d.html.*
15. A. Garg, R. Tamassia, and P. Vocca. Drawing with colors. In *Proc. 4th Annu. European Sympos. Algorithms (ESA '96)*, 1996.
16. D. Jablonowsky and V. A. Guarna. GMB: A tool for manipulating and animating graph data structures. *Softw. – Pract. Exp.*, 19(3):283–301, 1989.
17. B. Monien, F. Ramme, and H. Salmen. A parallel simulated annealing algorithm for generating 3D layouts of undirected graphs. In F. J. Brandenburg, editor, *Graph Drawing (Proc. GD '95). LNCS*, 1027:396–408, Springer-Verlag. 1996.
18. S. Mukherjea. Visualizing the information space of hypermedia systems. Technical report, Graphics and Visualization Center, GeorgiaTech.
19. S. P. Reiss. 3-D visualization of program information. *Graph Drawing (Proc. GD '94). LNCS*, 894:12–24, Springer-Verlag. 1995.
20. G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3D visualizations of hierarchical information. In *Proc. CHI'91*, pages 189–193.
21. R. Sollenberger and M. P. The effects of stereoscopic and rotational displays in a three-dimensional path-tracing task. *Human Factors*. 35(3), 483–500.
22. L. Spratt and A. Ambler. Using 3D tubes to solve the intersecting line representation problem. In *Proc. IEEE Symp. on Visual Lang., 1994*, pages 254–261.
23. K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical systems. *IEEE Trans. Syst. Man Cybern.*, SMC-11(2):109–125, 1981.
24. R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.
25. R. Tamassia, G. Di Battista, and C. Batini. Automatic graph drawing and readability of diagrams. *IEEE Trans. Syst. Man Cybern.*, SMC-18(1):61–79, 1988.
26. C. Ware and G. Franck. Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Transactions on Graphics*. to appear. *http://www.omg.unb.ca/hci/projects/hci-gv3D.html.*
27. H. D. Ware, C. and G. Franck. Visualizing object oriented software in three dimensions. In *CASCON '93*, pages 621–620, 1993. *http://www.omg.unb.ca/hci/projects/hci-gv3D.html.*

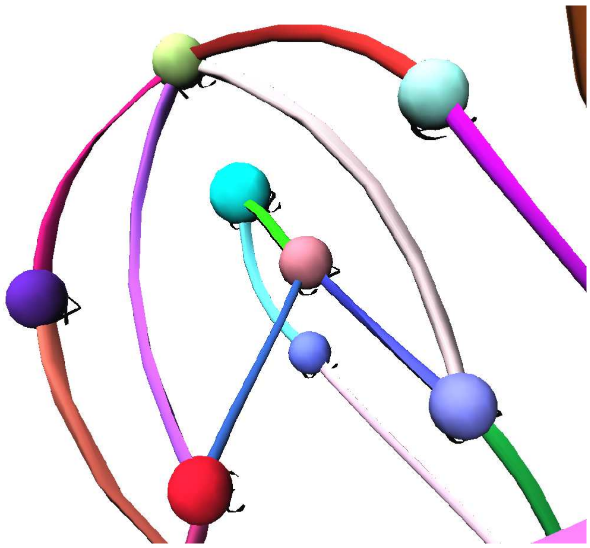


(a)

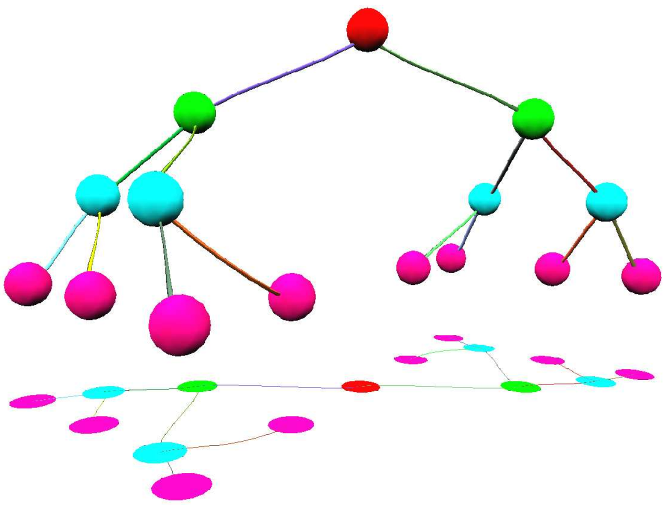


(b)

Fig. 2. Evolution History of UNIX



(a)



(b)

Fig. 3. (a) A Dynamic Model of World Interactions (from the book *World Dynamics* by Jay W. Forrester); (b) A Complete Binary Tree