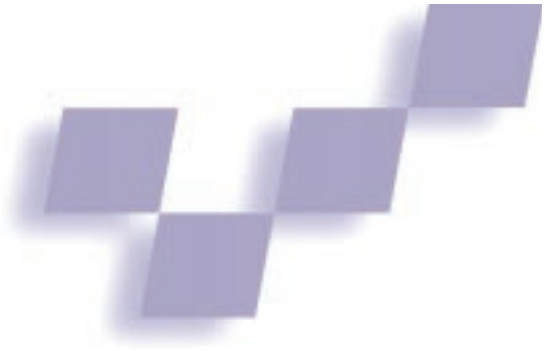


# TennisViewer: A Browser for Competition Trees

**Liqun Jin**
*Thomson Electronic Information Resources*

**David C. Banks**
*Mississippi State University*

A tennis novice watching a match for the first time might be surprised that the crowd erupts with cheers when a player wins one point, then barely applauds when he wins the next. The crowd is not necessarily fickle; some points are genuinely more important than others because a tennis match is hierarchically structured. One match consists of several sets. One set consists of several games. One game consists of several points. The match-winning point is the most important one.

How can we make that importance visible? Our goal is to let a fan, a player, or a coach examine tennis data visually, extract the interesting parts, and jump from one item to another quickly and easily. The visualization tool should help parse the elements of a match.

We developed an interactive system called TennisViewer to visualize the dynamic, tree-structured data representing a tennis match. It provides an interface for users to quickly explore tennis match information. The visualization tool reveals the overall structure of the match as well as the fine details in a single screen. It uses a 2D display of translucent layers, a design that contains elements of Tree-Maps[1] and of the Visual Scheduler system,[2] which was designed to help faculty and students identify mutually available (transparent) time slots when arranging group meetings. TennisViewer provides Magic Lens filters[3] to explore specialized views of the information and a time-varying display to animate all or part of a match.

## Competition trees

A tennis match is a tree-building process. When a player wins the last point of a game, he claims a node that lies one level higher—the game node. The game node inherits all his point nodes as its children. Both players always compete for the next higher available node, but only one of them can claim it. A player completes the tree by claiming the match node at the top level. The trees built by the two players together form a competition tree, which captures the hierarchical structure and the competitiveness of the tennis matches.

The Tree-Map offers a way to visualize hierarchically structured information with a $k$-D tree,[4] which is a general version of the orthogonal recursive bisection used in $n$-body simulations.[5] A Tree-Map depicts the hierarchy's structure and the content of the leaf nodes by using nested rectangles. We modified the Tree-Map so that each rectangle has a title bar to show the score in the corresponding node of a tennis match. The match node is a rectangle the size of the whole display. The region under its title bar subdivides vertically into subrectangles for sets. A set subdivides horizontally for each of its games. The process continues level by level, alternating the orientation of subdivisions at each level, to build a top-nesting layered map.

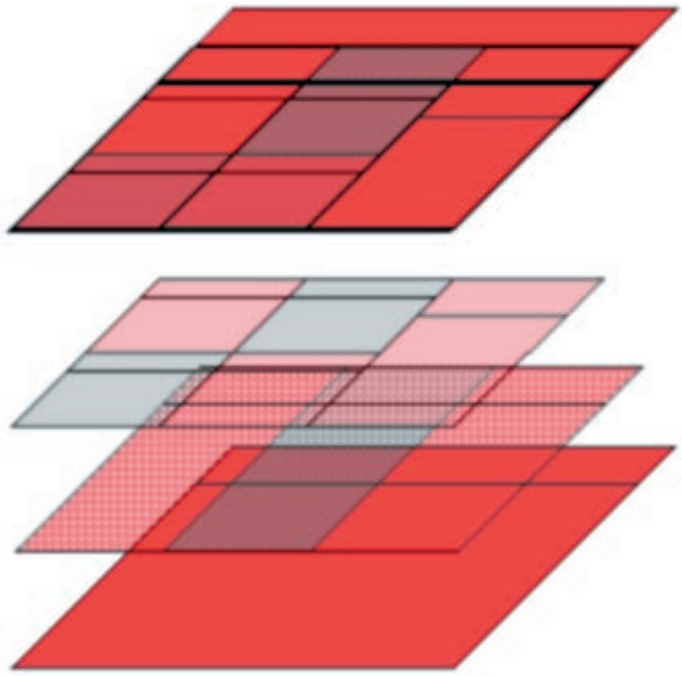
**TennisViewer employs competition trees to organize information on a tennis match. Magic Lenses magnify details, and an animated display depicts the dynamic nature of the game.**

Colors assigned to the rectangles represent node ownership: red nodes are owned by player one and green ones by player two, for example. When a player claims a node in the competition tree, his color is assigned to the corresponding rectangles. A child rectangle is layered over its parent rectangle so that its translucent color combines with the color of its parent, providing a more global context for the node. A red node in a red parent cell looks different than it does in a green parent cell. The former case corresponds to a point that contributed to a winning effort by player one. The latter indicates that the point was part of a losing effort.

The upper image in Figure 1 shows the result of blending the game layer and the set layer into the match layer at the bottom. The translucent layers provide local information (who won the point) with global context (who won the match).

Figure 2 shows the result of visualizing one simulated tennis match. J. Bond wins the match, 3 sets to 2. The color of the underlying match rectangle is red. Bond wins the first, third, and fifth sets, so their colors are red on red. M. Michel wins the second and fourth sets, so those rectangles' colors are green on red. When J. Bond

0272-1716/97/$10.00 © 1997 IEEE



**1** The top plane is constructed by projecting the two translucent planes onto the bottom plane.

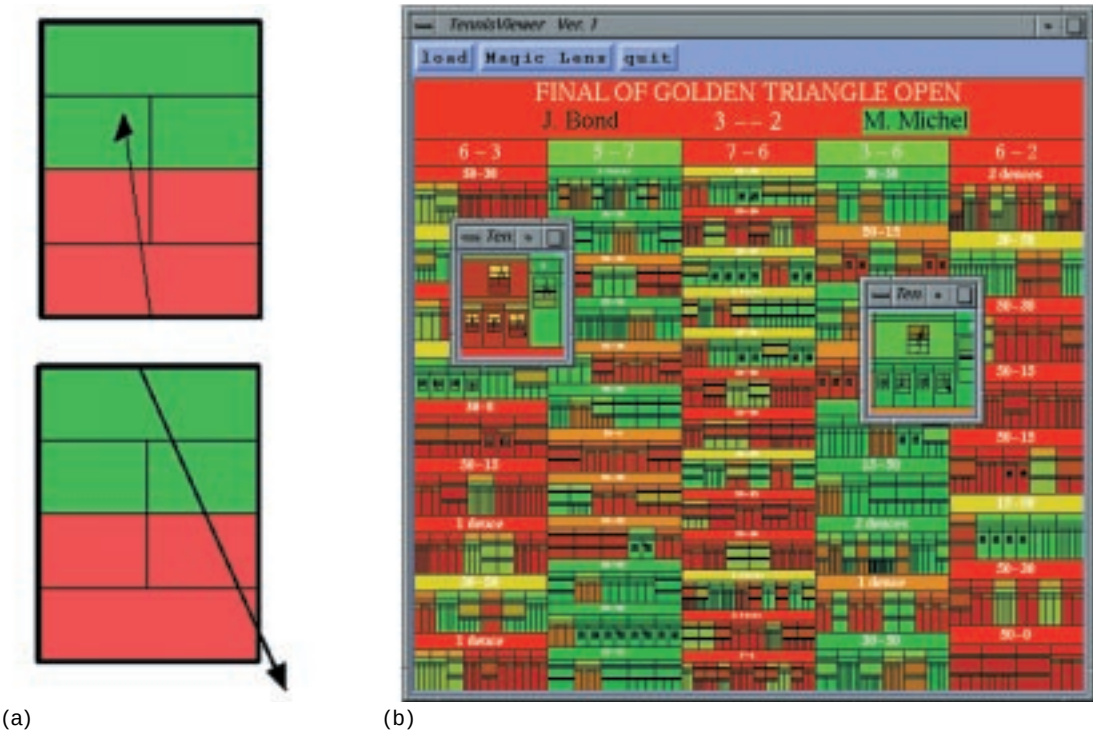
wins the match, all the information in the display becomes tinted by the global outcome, "winner = red," visible through the intervening layers.

We synthesized data for a tennis match between two players by simulating strokes subject to prescribed probabilities. The simulation demonstrates that the outcome of a match is very sensitive to the probabilities at the point level. For example, player one, who won 46 percent of the points in 10,000 simulated matches, only won 40 percent of the games, 24 percent of the sets, and 9 percent of the matches. A small departure from 50 percent probability at the point level produces increasingly exaggerated effects higher up the tree, a behavior that we wanted TennisViewer to reveal. The translucent layers produce this desired effect.

## Magic Lenses

To reveal the information at the low layers of the hierarchy, we must magnify the image. A fisheye view[6] is one method—it enlarges the image near the interesting parts. The Table Lens[7] uses the fisheye method to browse large tables. The Document Lens[8] uses the fisheye method to browse large documents. A Magic Lens opens a window over the image and re-displays the contents beneath it. By moving a Magic Lens on the image, the user can quickly explore the information underneath in greater detail.

We adopted Magic Lenses in our system to support exploring information at the lowest layers. The Magic Lens is a natural choice in a window system that supports rectangular viewports, and manipulating them is easy and fast. Moreover, Magic Lenses can easily be laid atop each other to produce deep zooming; for example, the tennis match information in Figure 2 includes one match, five sets, 51 games, 332 points, 446 serves, and more than 2,800 strokes (including the ball traces).




(a) (b)

**2** TennisViewer displays a computer-generated tennis match. (a) A serve (top) is returned out of bounds (bottom). (b) One Magic Lens filter lies on top of another, revealing ball traces within a point.

## Conclusion

We developed TennisViewer to give coaches, players, and fans a new way to analyze, review, and browse a tennis match. We chose tennis because of its popularity, because a tennis match includes so much information, and because tennis is organized in an obvious tree structure of points, games, sets, and match.

Translucent coloring of the nodes provides a multiresolution view showing which player won the various nodes. The user can quickly locate the interesting sets, games, points, services, and strokes, can determine the winners of each node according to colors, and can locate the long rallies based on the repeated occurrence of nodes belonging to a given player. Animation shows abrupt and global color changes after a crucial point (like winning a match point), which visually indicates the point's importance. The user can apply Magic Lens filters to explore the details of the match.

We envision the use of video at the map's lowest levels to permit an actual replay of specific parts of the match. Future work includes installing statistical lenses (like spreadsheets) to operate on the match data and lenses to show stroke details (top-spin, under-spin, volley) of interest to coaches, players, and fans.

TennisViewer runs under X/Motif on Unix platforms. We are working with ESPNet[9] to develop a Java version that runs in a Web browser to supplement conventional multimedia reporting of tennis matches. ■

### References


1. B. Johnson and B. Shneiderman, "Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures," *Proc. IEEE Visualization 91*, IEEE Computer Soc. Press, Los Alamitos, Calif., Oct. 1991, pp. 189-194.
2. D. Beard et al., "A Visual Calendar for Scheduling Group Meetings," *Computer Supported Cooperative Work (CSCW)*, ACM Press, New York, 1990, pp. 279-290.
3. E. Bier et al., "Toolglass and Magic Lenses: The See-Through Interface," *Proc. Siggraph 93*, ACM Press, New York, 1993, pp. 73-80.
4. J.L. Bentley and J.H. Friedman, "Data Structures for Range Searching," *ACM Computing Surveys*, Vol. 11, No. 4, Dec. 1979, pp. 397-409.
5. M. Berger and S. Bokhari, "A Partitioning Strategy for Nonuniform Problems on Multiprocessors," *IEEE Trans. on Computers*, Vol. C-36, 1987, pp. 570-580.
6. G.W Furnas, "Generalized Fisheye Views," *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems*, ACM Press, New York, Apr. 1986, pp. 16-23.
7. R. Rao and S.K. Card, "The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information," *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems*, ACM Press, New York, Apr. 1994, pp. 318-322.
8. G.G. Robertson and J.D. Mackinlay, "The Document Lens," *Proc. ACM Symp. on User Interface Software and Tech.*, ACM Press, New York, Nov. 1993, pp. 1-4.
9. *ESPNet*, http://espnet.sportszone.com.



***Liqun Jin*** *is a software engineer at Thomson Electronic Information Resource. He obtained his MS in computer science from Mississippi State University in 1996. He also received an MS and a BS in computer science from Fudan University, Shanghai, in 1989 and 1986, respectively. His main research areas include software engineering and visualization.*



***David Banks*** *is an assistant professor of computer science at Mississippi State University's NSF Engineering Research Center. His research interests include developing visualization techniques and user interfaces for studying multidimensional and time-varying data. He received a BS in mathematics from Davidson College, North Carolina, and an MS in mathematics and PhD in computer science from the University of North Carolina at Chapel Hill as a DEC Visualization Fellow. He received the NSF Career award in 1996.*

*Readers may contact Banks at the Dept. of Computer Science, Mississippi State University, 302 Butler, Mail Stop 9637, Mississippi State, MS 39762.*