# The Information Cube: Using Transparency in 3D Information Visualization

Jun Rekimoto *and Mark Green
Department of Computing Science, University of Alberta
Edmonton, Alberta, Canada T6G 2H1

December 5, 1993

## Abstract

In this paper, we propose a new 3D visualization technique for hierarchical information. This technique is based on the nested box metaphor, a familiar concept for any users. By using semi-transparent rendering, the system controls the complexity of the information presented to the user. With several 3D interaction techniques provided by the system, the user can recognize and inspect the information structure intuitively. Normally, the user puts on the VR equipment to use the system, but the system is also accessible through a conventional CRT display.

## 1 Introduction

Today, we must face many kinds of information which are difficult to represent in 1D or 2D, because they are too complex and do not fit into the current display's size and resolution. Examples of such information include large software structures, database schemes, organization structures, a huge project's PERT chart, traffic control information, and network management. All of them are huge, and it is quite hard to visualize them in one or two dimensional space without becoming lost in the detail.

As high performance graphics computers are getting popular, it becomes reasonable and practical to apply three dimensional display techniques to visualize complex information. However, although numerical data visualizations, which mainly attempt to visualize N-dimensional numeric data, have been well investigated under the name of "Scientific Visualization" for almost a decade, visualizing non-numerical information in 3D space, such as hierarchical or networked structures, is still a new research area and should be explored. Open problems we must address are as follows:

- Which visualization scheme is suitable for specific information structures?
- How do we navigate through a large information space?
- How do we interact with the information in 3D space?

In this paper, we propose a new 3D visualization technique of hierarchical information called the Information Cube. This technique uses translucent, nested cubes to represent hierarchical information. With the aid of three dimensional interaction techniques provided by the system, the user can manipulate and navigate through the information space quite effectively and intuitively.

The rest of the paper is organized as follows. In the next section, we summarize previous work about visualization of hierarchical information. The following two sections (Section 3, and 4) present our visualization technique and its user interface. Implementation issues regarding our system are discussed in Section 5. In Section 6, we discuss about our technique and other related issues.

## 2 Previous Work

In this section, we briefly summarize the previous work regarding information visualization.

Of course, there are many 2D tree layout algorithms have been proposed to date (for example, [6] or [2]). A common problem among these, however, is that the screen space is too limited to display a large tree structure. These 2D layout schemes force the user to see structures only partially, or the user must deal with the shrunk representation. Neither scrolling nor zooming solves this fundamental problem completely.

The desktop metaphor used in Macintosh Finder [4] can be regarded as a visualization of a hierarchical structure. The user can see the lower level information

*Current address: Sony Computer Science Laboratory Inc. 3-14-13, Higashigotanda, Shinagawa-ku Tokyo 141, Japan (rekimoto@csl.sony.co.jp).

by clicking the icon at the higher level (i.e., a window corresponding to the lower level appears on the screen). The problem of this method is that the user's desktop can easily be cluttered. In such a case, the user must look for the desired window by moving, resizing, rearranging, or closing other windows. The user often lost consciousness about which window belongs to which position in the hierarchy. Same information may appear in two ways – icon and window – simultaneously. No visual aids are provided to correspond these two representations.

Robertson et al. proposed a 3D visualization technique called the Cone Tree [11]. With this technique, hierarchical information is presented as 3D trees that look like cones. Each cone's apex corresponds to a node, and sub nodes are placed at the rim of this cone. Three dimensional perspective causes a natural fisheye effect; the user can bring an interesting sub tree to the front. Animation is also used to help the user to recognize information structure. This technique has clearly shown the effectiveness of 3D information visualization.

However, as the authors themselves pointed out in their paper, this technique becomes less effective when information is balanced, or when the nesting level is too deep. In these cases, the resultant image becomes too complex and less understandable even with the 3D perspective effect and interactive animation.

Another problem of this method is that its layout scheme is not truly three dimensional. Suppose that we have only two levels hierarchical information; one top-level node and 100 second level nodes. The resultant visual is a cone with top node at its apex and other nodes at its rim. Although the entire structure is visually understandable, to see every node in the second level, the user must rotate the cone 360 degree and check each node one by one. Such situation is essentially a rather expensive simulation of 1D item lists, such as Rolodex or Macintosh's list manager.

Johnson and Shneiderman's Tree-Map is another visualization technique of hierarchical structure [5]. Tree-Map represents hierarchical information as nested 2D boxes. Inner boxes, which correspond to lower level information, are stuffed in an outer box, which corresponds to higher level information. This technique uses screen space quite effectively, because neither additional graphics (e.g., the arc between a parent node and a child node) nor additional space (e.g., the space between a subtree and a subtree) is required.

Although having such advantages, its nested representation is hard to understand. The user must pay much attention to distinguish which box belongs to which level. As the number of levels or the number of items increases, resultant image becomes a very complicated. One reason of this problem is that the Tree-Map is too eager to utilize the screen space as the contents of information, thus the structural cue less appeals to the user. No redundancy of this method makes it difficult to the user.

# 3 The Information Cube

The Information Cube is a 3D visualization technique of hierarchical information. With this technique, information is visualized as nested cubes. The outermost cube corresponds to the top level data, while the next level data are represented as the cubes in the outermost cube. Each second level cube contains third level cubes, and so on. Each cube has title on its surface. Terminal data (which have no subsidiary data) are presented as tiles with labels on their surfaces.

The system displays these cubes in 3D. Either head-mounted display (HMD) or conventional CRT can be used to display the image. The user can rotate or move the cube with his/her hand (using a DataGlove), and can select one cube and navigate toward it for detailed inspection. Each cube is rendered in semi-transparent color, so the user can easily see the inside of the cubes.

Figure 1 is a snapshot of the system displaying a simple example which is used in [5]. Figure 2 is a more practical example which represents a Unix's directory hierarchy. Though this example contains about 1500 files and 50 directories, it nicely fits into the screen.

The metaphor used here is quite simple and natural. Virtually no effort is required for users to understand the meaning of the visualization, because we are quite familiar with the concept and the usage of a box – as a container – in our daily lives. In addition, a cube can contain arbitrary 3D objects. For example, Figure 3 shows a cube containing a 3D networked structure. In this way, the Information Cube is also a useful technique for organizing arbitrary 3D objects in a hierarchy.

## 3.1 Controlling Transparency

As shown in figure 1 and 2, the surfaces of the cubes are rendered with semi-transparent color. This transparency allows the user to see the inside, while hiding inner information gradually. If the surfaces are totally transparent (i.e., drawing cubes as wire frames), the image becomes too complicated to understand. On the other hand, if the surfaces are opaque, the user can only see the outermost cube; inner information becomes invisible. Semi-transparent rendering solves these two problems simultaneously. By using semi-transparent surfaces, the system can control the complexity of the information that appears on the screen. Even when the level of hierarchy is very deep, the complexity of the

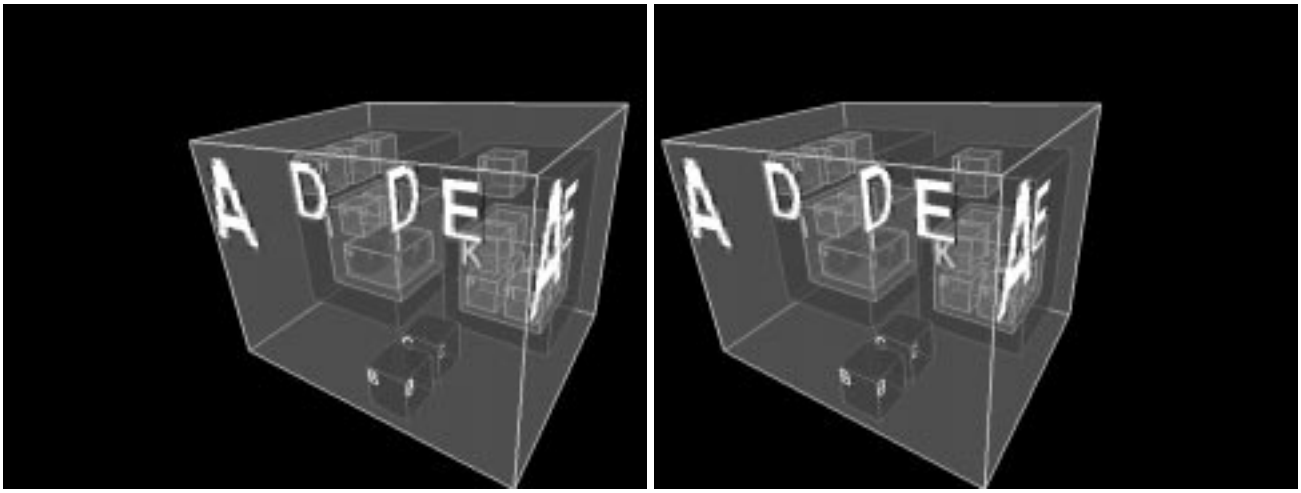

Figure 1: The Information Cube (Stereogram)

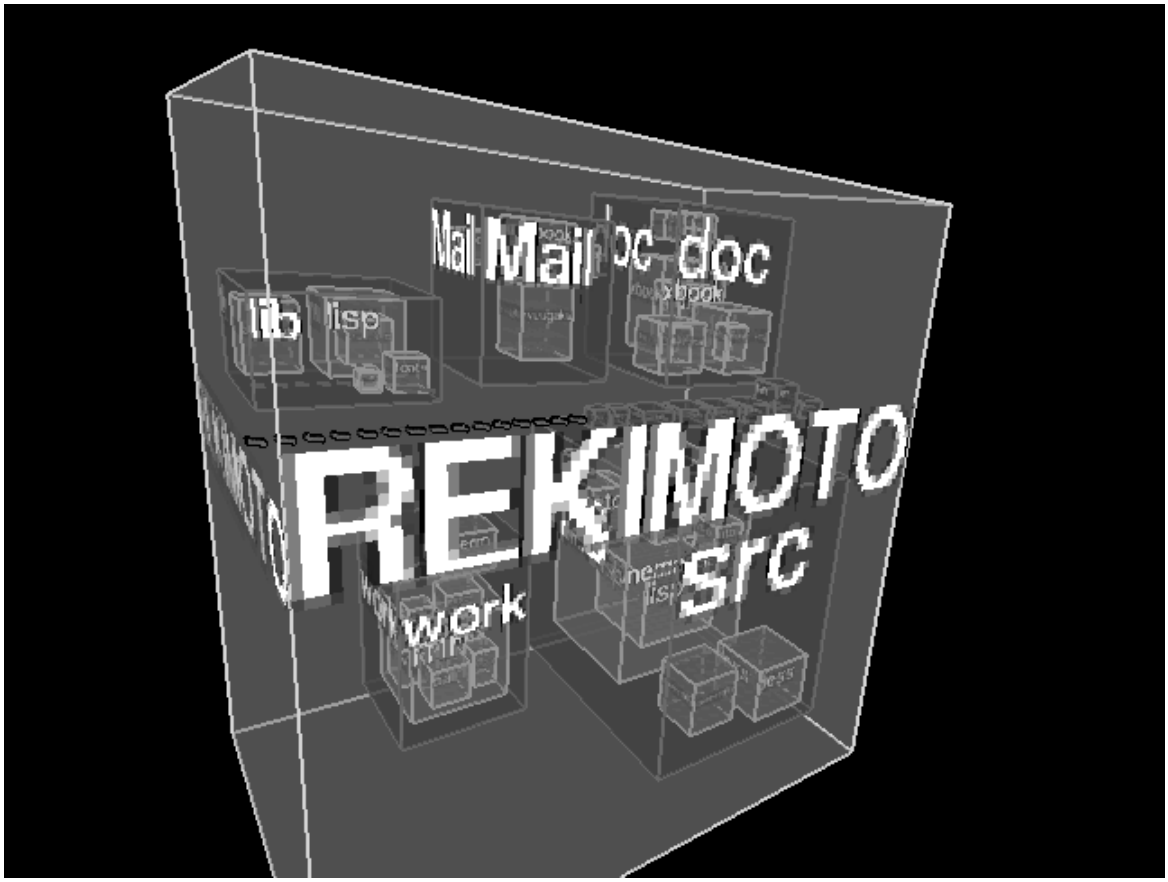


Figure 2: Displaying a Unix directory structure using the Infomation Cube

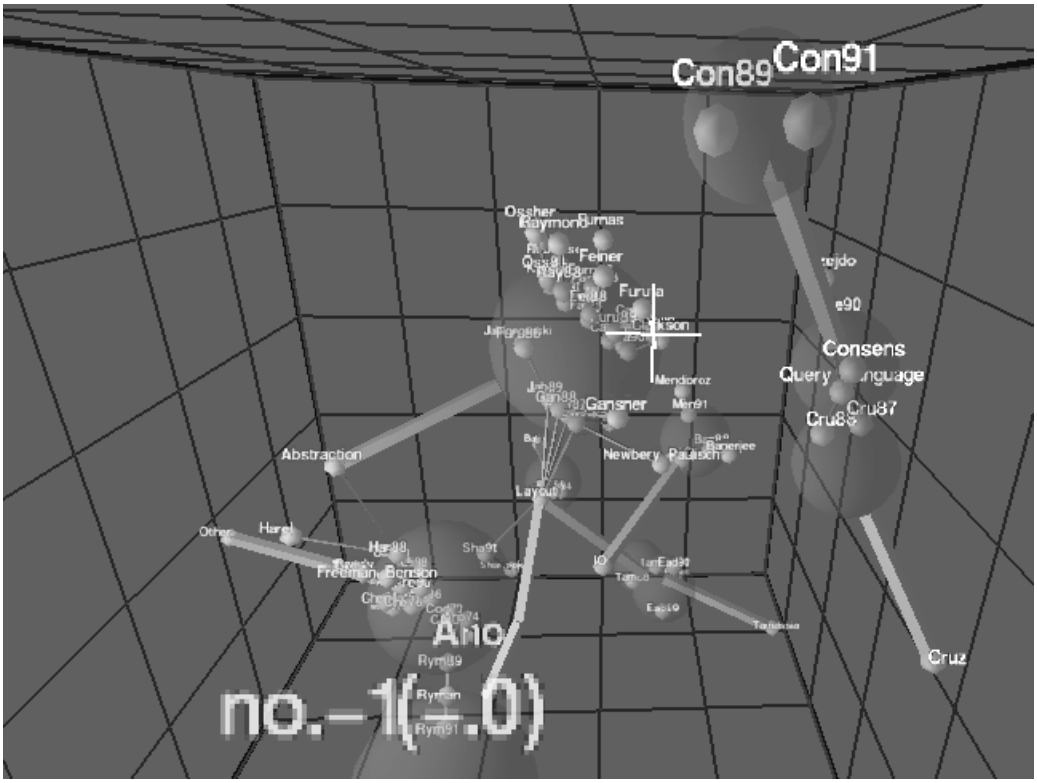


Figure 3: A cube containing a 3D graph

screen image is maintained reasonably, because inner level images are fogged out gradually.

Using transparency to represent a nested hierarchy is quite natural. The user can easily see which cube contains which cube. Semi-transparent surfaces separate higher and lower information clearly, without hiding the information. This feature makes possible for the user to see and manipulate complex hierarchical structures, because complexity of the screen image is always maintained at a reasonable level. In addition, semi-transparent surfaces can convey several kinds of additional information. For example, the system can represent accessibility, changing rate, or importance of the information by its transparency.

Though using transparency in 3D graphics is not new[1], we believe that applying transparency to represent information structures is a novel and useful idea. Another possible usage of transparency in information visualization is to represent a group (aggregation) of information. For example, Figure 3, which is a visualization of a network structure representing a document database, uses semi-transparent spheres to represent aggregated nodes (a node which is comprised of nodes). The user can see inside easily, without losing the notion of the group.

[1]In fact, volume visualization, for example, often uses translucent/transparent objects to represent numeric data.

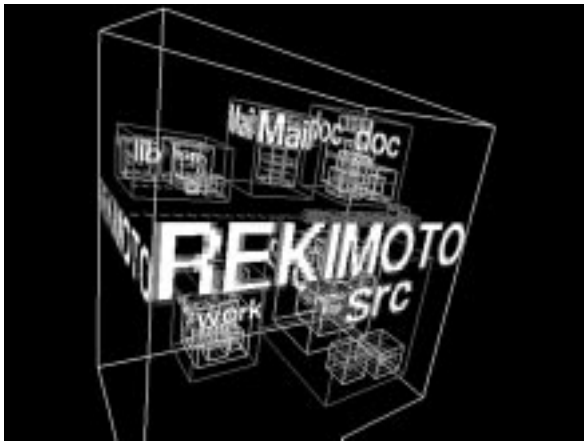


Figure 4: Same Information of Figure 2 without Semi-Transparent Rendering

### 3.2 Object Layout

Since the Information Cube represents a hierarchical information structure, cubes may contain sub-cubes. We need some kind of layout algorithm to determine the location of each cube.

There are several algorithms to pack sub-cubes into the parent cube. An optimal layout that minimizes the amount of spaces is known as "Knapsack Problem" or "Bin Packing Problem." However, we consider such optimal packing is not necessary in our system for several reasons. Optimal packing is quite time consuming (i.e., NP complete), so it is not suitable for interactive applications. When cubes are packed up too tightly, the user has difficulty to see the inside even if each cube is transparent. On the contrary, some redundancy helps the user; a space between cubes helps the user to look behind cubes.

According to the above reasons, current our implementation uses a simple two-path algorithm to layout cubes. The policy of this algorithm is as follows. During the first path, each cube's size is calculated by summing up subsidiary's size recursively. In the second path, the algorithm determines each cube's location.

To keep the total size of the cubes reasonable, we reduce the scaling factor for the inner levels. For instance, the outermost (top level) cube is scaled to 100%, while cubes belong to the next level are scaled to 90%, the third level is scaled to 81%, and so on. When the user focused on the specific level, the system automatically re-scales the cubes so that every level looks like to be scaled uniformly.

## 4 Interaction Techniques

To manipulate the information stored in the Information Cube, we use a DataGlove [15] as an interaction device. The DataGlove is mainly used for two purposes, rotation and selection.

### 4.1 Object Rotation

To give the user a good insight into the 3D structure, we provide a stereoscopic view (via EyePhone) and motion parallax (by tracking the user's head). Interactive rotation also gives a strong cue to the 3D structure, so the system also supports it. In daily life, people always examine an item in their hands by rotating it. Interactive rotation is also known to be effective in inspecting objects in 3D-CAD systems. In addition, this method can also be used to move an interesting object to the front.

While the user is making a "grab" gesture with the DataGlove, the orientation of the cube is directly connected to the orientation of the glove. The user can rotate the cube freely by rotating the hand. The user can also change the location of the cube by moving the hand. The interaction is quite natural and it seems as if the user is grabbing the cube with the hand. When the user releases the cube, the orientation and the location are determined.

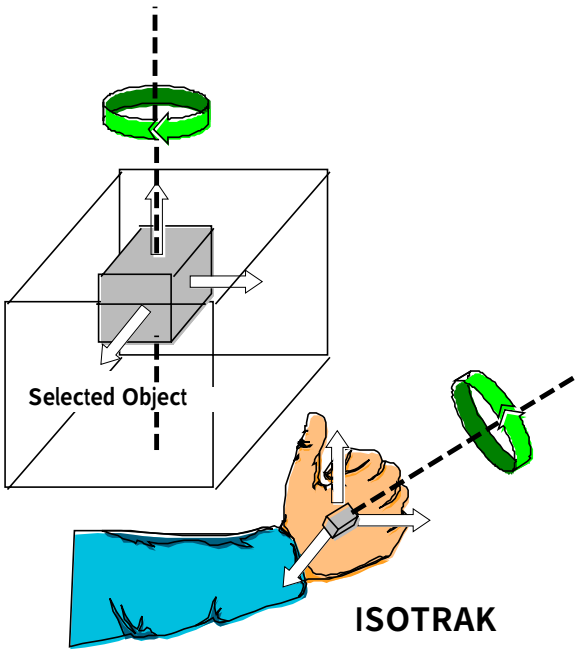


Figure 5: Rotating cubes

### 4.2 Object Selection

By using the Information Cube, the user can see the entire hierarchical structure from the outside; the semi-transparent surface allows the user to see inside the cube. However, when the user wants to inspect the

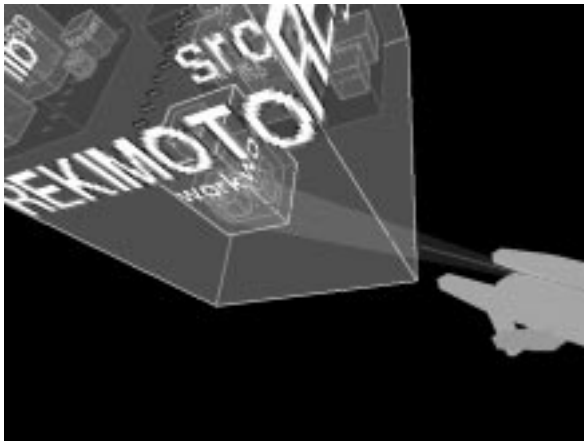

Figure 6: Object Selection

lower-level information more closely, the user needs to go inside the cube.

To accomplish this, we provide a *focus* operation, which is analogous to Unix's "cd" (change directory) command. To focus a cube, the user points at one of the neighborhood cubes and issues a "select" gesture. When the system selects the focused cube, the user moves toward the position at which the user can see the entire shape of the focused cube. Hence this method also works as a navigation method. The eye position after selection is close enough to inspect the inside the focused cube, but is not inside the cube so as to preserve the context of the selected cube in a hierarchy. Scaling is also applied automatically so that the user can see the small cube easily.

The translation and scaling do not occur suddenly. Instead, the system animates the transition smoothly, so that the user does not feel disorientated after changing the focus. To present this transition more naturally, the system controls the speed of the animation. We use *slow-in-slow-out* control, which has been extensively used in many cartoon films to animate a character's motion. With this velocity control, the initial and final speed is slow and intermediate speed is fast. The user feels that the motion is controlled by natural acceleration and deceleration.

The focused cube is visually enhanced by lowering other cubes' brightness and opacity slightly. The rotation center is also set at the center of the focused cube, so the user can turn the entire world around the focused cube.

Unlike Unix's "cd" command, the outside world of focused cube is still visible (because of its transparency), and the user can point outside cube to change focus. This feature prevents the user from losing context awareness [2]. As a result, navigating around the hierarchy is much smoother than that of Unix's cd.

As a pointing method, we use an interaction technique called the *spotlight selection*, which was invented by Liang as a selection method for 3D-CAD systems[7]. With this method, a virtual ray is emitted from the user's hand along the orientation of the hand. When the ray hits an object, it becomes a candidate for selection. The user confirms the selection by making a "select" gesture. To make small or distant objects easier to select, a virtual spotlight is also emitted along the ray. When the ray does not hit any object, then objects that touch the spotlight (which is a cone) become candidates for the selection. If more than one object touches the cone, a variant of Liang's anisotropic metric is used to decide which object becomes the candidate.

[2] Zuo et al. claims similar advantage in their 2D visualization system for hierarchical network structures [16].

# 5 Implementaion Issues

The Information Cube is implemented on top of the Mimimal Reality (MR) Toolkit[3, 12] with the IRIS GL graphics library [13]. The MR Toolkit is a platform system for Virtual Reality (VR) applications, which is developed at the University of Alberta.

Although the Information Cube's implementation uses high-end graphics capabilities such as texture mapping and alpha blending, more than 1000 nodes can be displayed in real time. For instance, the Unix directory example (see Figure 2), which contains 1500 files and 10 levels, can be displayed at approximately 8 frames per second on the Silicon Graphics IRIS 4D/VGX workstation.

In order to gain performance, the system does not display all data but omits some lower level information (e.g., labels of lower level leaf objects are not displayed). The fogging out effect of transparent rendering makes such omission less noticeable to the user.

## 5.1 Semi-Transparent Rendering

In general, drawing translucent 3D objects is not a trivial task. The implementer must control the order of rendering correctly for himself, because Z-buffer does not help when rendering transparent objects. There are several studies on this subject such as [8]. Some volume rendering systems generate semi-transparent objects by using ray-tracing algorithm. Though its results are very fine, ray-tracing is too cost intensive and not suitable for interactive applications.

In our case, however, cubes may be nest, but never intersect each other. By this reason, we can determine the order of drawing by applying depth sort recursively for each level of hierarchy. If cubes are drawn in right order, Z-buffer also works well for translucent objects, as well as solid objects.

Labels on the cubes are also translucent. To achieve translucent character rendering in 3D, the system uses texture mapping. The system prepares one texture data (currently, we use $256 \times 256$ pixels) that contains all characters's shapes (glyphs), and this data is used as a character generator. To draw each character on the specific surface at the specific transparency, the area in the texture data which contains one character's glyph is mapped onto the surface of the cube. This technique achieves fast rendering of semi-transparent characters with arbitrary size, orientation, intensity, and color.

## 5.2 Display Devices

Our system supports both a head mounted display (EyePhone) and a conventional CRT. The user can select one of them at a startup time. In either case, the

DataGlove is used as an input device.

While using CRT, another 6D input device (3SPACE Isotrak[9]) is used to change the position and orientation of the eye position (this is similar to the *eyeball in hand* metaphor, which is described in [14]). Normally, the user puts on the DataGlove on his right hand, while the left hand controls the eye position.

# 6 Discussions

In this section, we discuss the effectiveness of the Information Cube.

## 6.1 Applications

Currently, we are testing the effectiveness of Information Cube with three simple applications; they are the Unix directory browser, the C++ class hierarchy browser, and the Usenet news group browser. All of them share the same visualization program, and only data retrieval parts that extract hierarchical information from actual data are different.

Another experimental example we have developed is the multiple cube world which represents an "information city" where every building in the city is some kind of information. In this example, several kinds of information described above are represented as cubes and are placed on a large flat area. The user can fly by or walk around this area and can approach to an interesting data object. Since every cube is represented as semi-transparent, the user can see inside even from the distance. To move around large data space, a 3D navigation method called the NaviGrid[10], which is also developed by us, can be used.

Possible other applications include visual interfaces to information retrieval systems such as Gopher, a large organization chart, and an electric library database.

## 6.2 Comparison to other systems

One clear advantage over Robertson et al's Cone Tree [11], is that the Information Cube can contain a large number of childres objects, because its layout scheme is truly three dimensional. For example, even 1000 sub objects can be placed as a $10 \times 10 \times 10$ grid and its visualization still looks reasonable. On the other hand, placing 1000 objects at the rim of the cone yields an unnaturally spreading shape.

Semi-transparent surfaces separate higher and lower information clearly, without hiding the information. This feature makes possible to the user to see and manipulate very huge and complicated hierarchical structure, because complexity of the screen image is always maintained reasonable.

Even for a balanced hierarchical structure, where the ConeTree generates a less understandable image, the Information Cube can produce an asymmetric image so that the user can recognize the structure easily. This feature is further enhanced by the interactive rotation that the system also provides. Moreover, the image generated by the system is unique to the information's structure, because the size of the cube roughly reflects the amount of data being contained, and their aspect ratios are not always equal. This unique scene gives a user a good clue for remembering where the information is stored in a hierarchy.

## 6.3 Interaction

Interactive rotation certainly helps the user to recognize the information structure. Based on our experience, this dynamic cue is much stronger than the static stereoscopic cue.

One trick used here is that we restrict the rotation about only Z-axis (vertical axis). In our early implementation, we allowed free rotation (in any directions) by the hand, which caused disorientation to the user. The user tended to be confused about the positional relationship among cubes. We think that allowing too many degrees of freedom sometimes confuses the user. To eliminate this problem, we only use the rotation about the axis of the forearm and connect it to the cube's Z-axis. Other elements of rotation are simply ignored. The result is quite successful.

This restriction is also adequate from the human ergonomics point of view. The rotation about the forearm is much easier than other rotations. It also helps the user to control the orientation and the position simultaneously. Since the system ignores rotation about axes other than the object's Z-axis, the user can take a natural posture to do rotation.

## 6.4 HMDs vs. CRTs

Since our system both supports HMD and CRT, we could compare HMD and CRT as a display device for information visualization.

We must admit that the resolution of the current HMDs is not enough for information visualization. For example, the number of pixels of HMDs is only about 1/20 of those of CRTs used with popular workstations. Color resolution of HMDs is also not good, that makes hard for the user to recognize subtle transparent objects, even if they appear on the CRT quite clearly.

Regarding interaction, however, we also felt a great potential of HMDs. HMD's head-coupled, binocular images certainly make manipulation easier. Especially, when the user selects an object by his hand (with the spotlight selection), we observed the HMD helps the

user, because this task requires a correct recognition of the position and the orientation of the spotlight in the 3D space.

# 7 Conclusion

In this paper, we propose a new 3D visualization technique for hierarchical structures. New ideas used in this technique are

- the nested box metaphor for visualizing large hierarchical structures. which is natural and familiar.
- semi-transparent rendering, which controls the complexity of the information on the screen.
- 3D interactive techniques for navigating and manipulating the information space.

We actually implemented a 3D visualization system called the Information Cube, based on ideas described in this paper. Informal experience shows this technique is effective and usable. We observed that the interactivity (such as rotation and zooming) provided by the system particularly enforced the user's understanding. We strongly believe that a 3D visualization coupled with effective interactive techniques is the promising area of future information systems.

# Acknowledgment

We would like to thank Chris Shaw for his suggestions and comments on this work. Jiandong Liang gave us various insights about 3D interaction techniques. Dani Beaubien and other colleagues at the Computer Graphics Laboratory in the University of Alberta supported the first author while he was studying the MR Toolkit. Finally, we would like to appreciate the University of Alberta and NEC Corporation for their support during the first author's stay at the Computer Graphics Laboratory.

# References

[1] Kim M. Fairchild, Steven E. Poltrock, and George W. Furnas. *SemNet : Three-Dimensional Graphic Representations of Large Knowledge Bases*, pages 201–233. 1988.

[2] E. R. Gansner, S. C. North, and K. P. Vo. DAG – a program that draws directed graphs. *Software – Practice and Experience*, 18(11), Nov. 1988.

[3] Mark Green and Dani Beaubien. *Minimal Reality Toolkit Version 1.2 Programmer's Manual.* Department of Computing Science, University of Alberta, Edmonton, Alberta, 1992.

[4] Apple Computer Inc. *Macintosh User Interface Guidelines.* Addison Wesley, 1992.

[5] Brian Johonson and Ben Shneiderman. Tree-Maps: A space-filling approach to the visualization of hierarchical information structures. In *IEEE Visualization '91*, pages 284–291, 1991.

[6] Tomihisa Kamada. *Visualizing Abstract Objects and Relations.* World Scientific, 1989.

[7] Jiandong Liang and Mark Green. Geometric modeling using six degrees of freedom input devices. In *Proc. of the 3rd International Conference on CAD & Computer Graphics (CAD/Graphics '93)*, Beijing, China, 1993.

[8] A. Mammen. Transparency and antialiasing algorithms implemented with the virtual pixel map technique. *CG&A*, 9(4):43–55, July 1989.

[9] Polhemus, Inc., Colchester, Vermont. *3SPACE ISOTRAK User's Manual*, 1987.

[10] Jun Rekimoto and Mark Green. The navigation grid: An effective technique for virtual space navigation. In *Proceedings of the Fifth Annual Western Computer Graphics Symposium*, 1993.

[11] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone Trees: Animated 3D visualizations of hierarchical information. In *Proceedings of ACM CHI '91*, pages 189–194, 1991.

[12] Chris Shaw, Jiandong Liang, Mark Green, and Yunqi Sun. The decoupled simulation model for virtual reality systems. In *CHI '92 Conference Proceedings*, 1992.

[13] Silicon Graphics, Inc., Mountain View, California. *Graphic Library Programming Guide*, 1991.

[14] Colin Ware and Steven Osborne. Exploration and virtual camera control in virtual three dimensional environments. In *Proceedings of the 1990 Symposium on Interactive 3D Graphics*, pages 175–183, 1990.

[15] T.G. Zimmerman and J. Lanier. A hand gesture interface device. In *Proceedings of CHI and Graphics Interface 1987*, pages 189–192, 1987.

[16] Zhengping Zuo, John Dill, and Lyn Bartram. Variable Zoom: A fisheye view of hierarchical networks. In *Proceedings of the Fifth Annual Western Computer Graphics Symposium*, 1993.