

# Comparing Fisheye and Full-Zoom Techniques for Navigation of Hierarchically Clustered Networks

Doug Schaffer\*, Zhengping Zuo<sup>¥</sup>, Lyn Bartram<sup>¥</sup>, John Dill<sup>¥</sup>,  
Shelli Dubs<sup>†</sup>, Saul Greenberg\*, Mark Roseman\*<sup>1</sup>

\* Dept of Computer Science  
University of Calgary  
Calgary, AB, T2N 1N4 Canada  
schaffer,saul@cpsc.ucalgary.ca  
+1-403-220-6087

<sup>†</sup>Alberta Research Council  
6815 8St NE  
Calgary, AB, Canada

<sup>¥</sup>School of Engineering Science  
Simon Fraser University  
Burnaby, BC, V5A 4S6 Canada  
dill@cs.sfu.ca,  
+1-604-291-3574

## ABSTRACT<sup>1</sup>

Many information systems are represented as two-dimensional networks of links and nodes. Because these networks tend to be large and quite complex, people often prefer to view part or all of the network at varying levels of detail. *Hierarchical clustering* provides a framework for viewing the network at different levels of detail by superimposing a hierarchy on top of it. Nodes are grouped into clusters, and clusters are themselves placed into other clusters. Users can then navigate these clusters until an appropriate level of detail is reached.

This paper describes an experiment comparing two methods for viewing hierarchically clustered networks. Traditional *full-zoom* techniques provide details of only the current level of the hierarchy. In contrast, *fish-eye views* (generated by the “variable zoom” algorithm described in this paper) provide information about higher levels as well. Subjects using both viewing methods were given problem solving tasks requiring them to navigate a network, in this case a simulated telephone system, and to reroute links in it. Results suggest that the greater context provided by fish-eye views significantly improved a user’s performance of the tasks. They were quicker to complete their task, and they made fewer unnecessary navigational steps through the hierarchy. This validation of fish-eye views is important for designers of interfaces to complicated monitoring systems, such as control rooms for supervisory control and data acquisition systems, where efficient human performance is often critical.

**KEYWORDS:** Fisheye views, information visualization, hierarchically clustered graphs, SCADA interfaces.

## 1 INTRODUCTION

*“He couldn’t see the forest because of the trees...”*

People naturally perceive the world using both local detail and global context. Biologically, our eyes let us see detail for only a small focused region. Yet we remain constantly

aware of the global context through our peripheral vision and by glancing around. We rely heavily on global context to orient ourselves and to understand local detail; indeed, tunnel vision (which can be simulated by holding a paper tube to one’s eye) is considered a serious handicap.

Most computers naturally encourage tunnel vision interfaces, for they supply users with very small screens to view large complex information spaces (even a 19” display consumes only a fraction of our normal field of view). Interfaces can minimize the tunnel vision effect through several fundamental strategies. First, traditional graphical systems supplied *pan and zoom* capabilities, where users can pan or scroll a window across a virtual canvas, and they can adjust the scale of their view (and the entire space) through zooming. The problem is when users are zoomed out for orientation, there is not enough detail to do any real work. When they are zoomed in sufficiently to see detail, context is lost. Second, *multiple windows* may be provided, each with a pan and zoom capability. While reasonable for small information spaces, the many windows required by large spaces often lead to usability problems due to excessive screen clutter and window overlap. Third is the *map view* strategy, where one window contains a small overview, while a second window shows a large more detailed view (Beard and Walker 1990). The overview contains a rectangle which can be moved and resized, and its contents are shown at a larger scale in the large view. Map views suffer from the extra space required for the overview, and from forcing the viewer to mentally integrate detail and context.

Recent advances in computer-based information visualization have acknowledged the importance of balancing local detail with global context into a single view by providing *fish-eye views* of the data space (Furnas 1986). Analogous to a wide-angle camera lens, the idea is to show “local” detail in full (the objects of interest to the user), while displaying successively less detail for information further from a focus of attention. This can be done by three methods. First, we can graphically distort the view, where items shrink as they move away from the focus point. Second, we can present partial views through filtering, where a distance function determines whether or not items

<sup>1</sup>Authors from Simon Fraser designed and built the fish-eye view system, while authors from the University of Calgary and the Alberta Research Council conducted the study and subsequent analysis.

should appear on the display. Finally, we can use simpler, smaller representations and abstractions, for example, representing a detailed circuit as a rectangle or icon.

Particular fisheye techniques for viewing large information spaces (graphs) have been proposed and implemented by several researchers. Furnas (1986) described a generalized “degree of interest” function where the interest value of a node in the graph is a function of both its a priori importance and its distance from the user's current focus. He created systems for viewing and filtering structured program code, biological taxonomies and calendars and verified that fisheye views were indeed superior to flat views by performing a modest usability study. Sarkar and Brown (1992) pursued a mostly graphical approach to fisheye views. These included planar and polar transformations of connected graphs, and used Euclidean distance to calculate degree of interest. Their system shows all nodes within the network, unless a particular node's “display” value fell below a threshold, in which case it was removed from the graph's view. While the resulting images are impressive, they do note that users sometimes perceived the resulting view as unnatural. Mackinlay, Robertson and Card (1991) linearly transformed a 1-d space by projecting it on a 3-d “Perspective Wall”, while (Robertson, Mackinlay and Card 1991) combined 3-d effects and animation for displaying hierarchies in “Cone Trees”. Johnson and Shneiderman (1991) used a space-filling algorithm to fit a complete strict hierarchy into a window (see also Shneiderman 1991). It is based upon every node containing a value that is the sum of the node values of its children. This value determines the node's relative size on the screen. Their system, called “TreeMaps”, displays a hierarchical file system, where the value shown is the size of the directories and files. Tree Maps presents a very different way of viewing information, and there are still outstanding questions on its usability. Finally, Schaffer and Greenberg (1993) promoted ideas of information filtering and fisheye views of hierarchies through the use of dynamic queries.

While there is much interest and intuitive appeal in fisheye views, there have been few quantitative studies evaluating its merits. Our own work applies a particular kind of fisheye view, which we have called “variable zoom”, to hierarchically clustered networks (explained in Section 2). These can be used to represent and view real environments—telephone systems, oil pipelines, power grids—that are controlled by operators of Supervisory Control and Data Acquisition (SCADA) systems. Because these are critical environments, we wanted to see how well people could navigate and manipulate a hierarchical graph using either a traditional zoom method or fisheye views.

We begin the paper with a brief description of the Simon Fraser variable zoom display algorithm, which was used to provide a fisheye view interface to a simulated telephone network. We then describe a controlled experiment contrasting user performance using both a standard full-

screen zoom view and a fisheye view. After presenting the results, we re-examine the experiment and the system, and then suggest several implications our research has to interface design.

## 2 THE VARIABLE ZOOM DISPLAY METHOD FOR TWO-DIMENSIONAL NETWORKS

Furnas' fisheye approach was quite effective for tree structures; our work extends his ideas to 2-d graphs with superimposed *hierarchical clustering* (Fairchild, Poltrok and Furnas 1988) and supports multiple foci. Our method, as in Sarkar and Brown (1992), deals with a connected graph as the primary data structure. Other researchers have developed various kinds of hierarchical graph structures such as Higraphs (Harel 1988) and Hypergraphs (Berge 1973) to which the hierarchical clusters used here bear a resemblance. However, our interests lie in ways of graphically representing the structure that are of help to a user, rather than the underlying graph-structure properties of the base network.

The variable zoom method works upon a 2-d network of nodes and links, such as the one shown in Figure 1a. It assumes that a hierarchical clustering of nodes has been superimposed on the network. Figure 1b, for example, shows how nodes in the network of Figure 1a have been clustered into three hierarchical levels. The largest rectangle is the “root” cluster of the hierarchy, and contains five smaller clusters a-e (the second level of the hierarchy). These in turn may or may not contain other clusters, nodes, or combinations thereof. The bottom of the hierarchy is reached when a cluster contains only network nodes. As long as a strict hierarchy is maintained, nodes can be clustered in any way the designer wishes, e.g., by using geographical distance, by task-specific relationships of nodes, and so on.

The hierarchy is then used by the visualization algorithm to allow clusters of the network to be viewed at different levels of hierarchical detail. At each level above the network node level, we represent the clusters as icons that may be “opened” to show the next level down. In Figure 2a, for example, clusters a through e are drawn as icons; the links indicate that clusters are connected by at least one path in their respective sub-nets. Figure 2b shows the operation of opening (zooming into) two higher level icons (a and d). The key to obtaining the fisheye effect is to uniformly magnify appropriate parts of lower levels to show detail while embedding this detail in the remaining, uniformly scaled down, network. An advantage of this method is that multiple areas of focus (detail) are allowed.

The algorithm assumes all nodes (leaf and cluster) are square and do not overlap. That is, projections of nodes on x and y axes do not overlap<sup>1</sup>. It works by applying the

---

<sup>1</sup>Extending the algorithm to allow overlap and to rectangles is relatively straightforward.

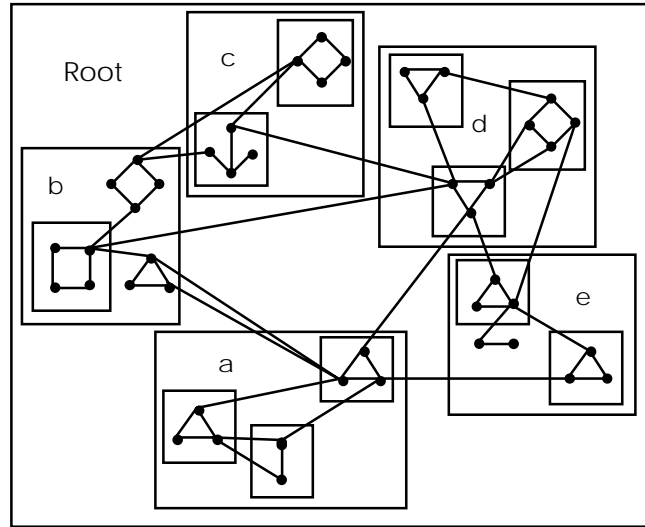
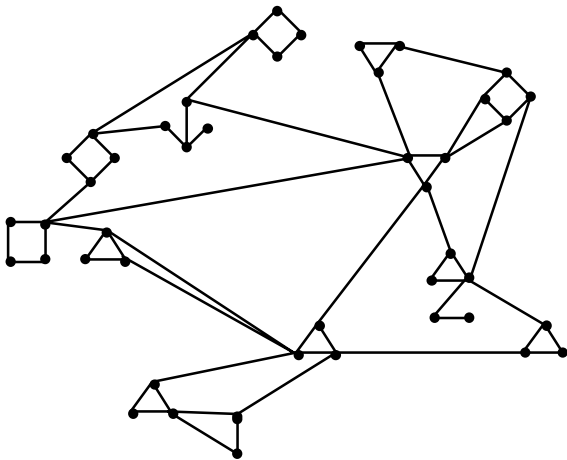
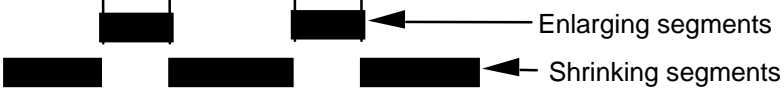
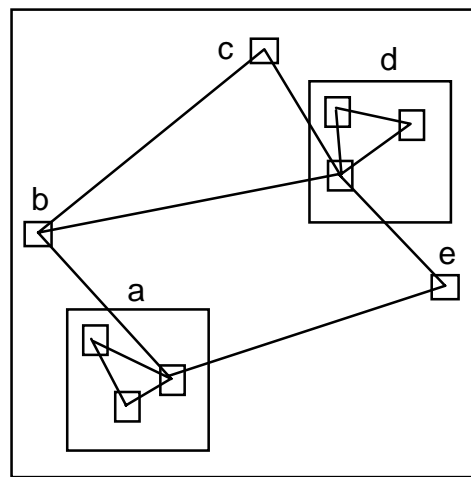
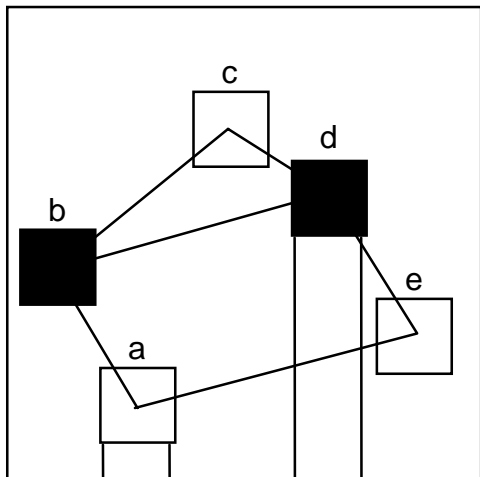


Figure 1. a) an example network, and b) an example of how the network can be hierarchically clustered.



(a)

(b)

Figure 2. Example of basic operation. (a) network before zooming. (b) nodes 'a' and 'd' have been zoomed to show their subnetworks; other parts are shrunk as context.

same method in horizontal (x) and vertical (y) directions; the description below applies to both. Nodes are in either a zoomed or unzoomed (iconicized) state: zoomed nodes are opened like windows, displaying their immediate subnetwork. Unzoomed nodes are closed as icons; their subnetworks are not shown. Cluster links are simply straight lines joining clusters.

In the actual system, both network nodes (leaves) and cluster nodes are represented as squares. Texturing is used to visually distinguish the two. Texturing is also used to differentiate lines representing two or more real links in the network from those representing single links. Texture is displayed using white, vertical stripes on the background color node color.

The algorithm is described for a single node, assumed to be in a zoomed state, and its subnetwork. The display size of each node or cluster in the subnetwork is calculated. Nodes and clusters to be magnified are zoomed by a magnification factor,  $F_e$ , and others are reduced in size by a shrink factor,  $F_s$ . Finally, placement of all nodes and clusters is calculated. The resulting procedure is then applied recursively to each cluster.

## 2.1 Magnification Factors

We first calculate  $F_e$ , the magnification factor to be applied to nodes to be zoomed, and  $F_s$  the shrink factor to be applied to the remaining spaces. We do this by considering two ratios.  $R_z$  is the ratio of nodes to be zoomed with respect to their environment (length of parent node,  $L$ ), and  $r$  is the ratio of nodes to be zoomed to the total length of all nodes, after the zoom operation is applied.

$$R_z = F_e S_z / L \quad 0 < R_z \leq 1 \quad (1)$$

$$r = S_z / S_a \quad 0 < r \leq 1 \quad (2)$$

where  $S_z$  = sum of lengths of all nodes to be zoomed, and  $S_a$  = sum of lengths of all nodes. The use of ratios keeps the development independent of the particular level in the overall network.

To make the subnetwork detail visible,  $R_z$  should never be smaller than some threshold value. Since a node may be arbitrarily small (and since the user may wish to zoom just one node),  $r$  can be arbitrarily small, and we need

$$R_z \geq \text{threshold as } r \rightarrow 0 \quad (3)$$

If all nodes are zoomed ( $r = 1$ ), no context need be retained, and

$$R_z = 1 \quad \text{when } r = 1. \quad (4)$$

As the number of nodes increases from one to the total number (in general as  $r$  increases from zero to one),  $R_z$  should increase from the threshold to one smoothly and monotonically. A simple relationship meeting all these

requirements is:

$$R_z = k_1 r + k_2$$

where  $k_1$  and  $k_2$  are constants. From (4),  $k_1 = 1 - k_2$ ; renaming  $k_2$  as  $K_b$  (a balance factor, discussed below), we have

$$R_z = (1 - K_b) r + K_b \quad (5)$$

and substituting into (1),

$$F_e = K_b (L/S_a) (1/K_b - 1 + 1/r). \quad (6)$$

To use space effectively, the sum of magnified and demagnified segments should equal the length of the containing node after the operation, so that

$$F_e S_z + F_s (L - S_z) = L, \text{ and} \\ F_s = (1 - F_e S_z/L) / (1 - S_z/L) \quad (7)$$

The expressions for  $F_e$  and  $F_s$  indicate both are dependent on the environment and on the user's request (i.e. on the number of nodes to examine in detail). For this reason we refer to the algorithm as a "variable zoom" type of fisheye method.

## 2.2 Basic Operation

The operation applied to each node recursively simply calculates the new sizes and locations. The sizes are just the original sizes multiplied by  $F_e$  or  $F_s$  as appropriate. To calculate the positions, the x-axis is divided into segments by the boundaries of nodes to be zoomed (an identical procedure is used on the y-axis). Segments corresponding to nodes to be zoomed are enlarging segments; the others are shrinking segments (Figure 2a). Let  $x_i$  and  $x_i'$  be the positions before and after,  $l_s$  the length of the segment, and  $d_i$  be the distance from  $x_i$  to the left boundary of the segment containing  $x_i$ . The  $x_i$  are calculated by first sorting the segment list from left to right, and then performing the following for each node (the result is shown in Figure 2b).

```

initialize  $x_i'$  to the left boundary
of parent node
for each segment to the left
of  $x_i$ 
  if enlarging,
     $x_i' = x_i' + F_e l_s$ 
  else
     $x_i' = x_i' + F_s l_s$ 
for the segment containing  $x_i$ 
  if enlarging
     $x_i' = x_i' + F_e d_i$ 
  else
     $x_i' = x_i' + F_s d_i$ 

```

## 2.3 Balance Factor

The constant  $K_b$  in the expressions for  $F_e$  and  $F_s$  is a

“balance” factor that controls the ratio of detail area to parent area, i.e. the ratio of detail to context. A larger  $K_b$  gives a larger proportion of detail. To see this, we rearrange (5) as

$$R_z = r + (1-r) K_b$$

so that for a constant number of nodes to be zoomed ( $r$  is constant),  $R_z$  grows with  $K_b$ . Figure 3 illustrates the visual effects of varying  $K_b$ . Adjusting  $K_b$  thus allows a user to control the relative emphasis on detail and context.

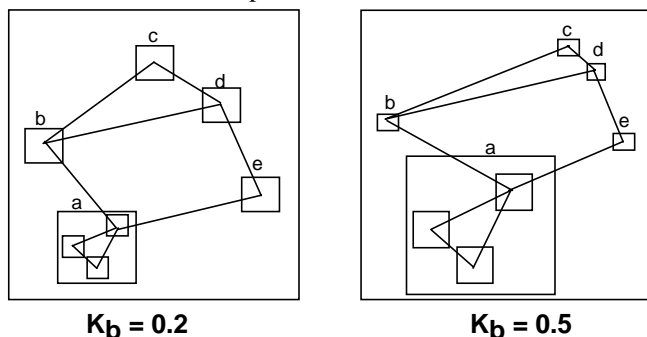


Figure 3. The visual effects of varying the balance factor.

### 3 DESCRIPTION OF THE EXPERIMENT

This experiment compared the performance of subjects navigating and repairing a simulated telephone network, represented as a hierarchically-clustered graph. Subjects use either a full-zoom or a fisheye view method to navigate the clusters.

*Hypothesis.* Our null hypothesis was that there is no difference in performance ( $p = .05$ ) between subjects traversing a hierarchically clustered network using a fisheye view or a full-zoom view. Since this is a within-subject experiment where each person uses both viewing methods, we counter-balance ordered the methods and propose a secondary hypothesis that order (and learning) have no effects on the outcome. Performance was measured in terms of the total time taken to complete the task, number of zooms performed, and their success at performing an assigned task.

*Subjects.* A total of 20 subjects were selected from a pool of volunteers. All were senior undergraduate students, graduate students, or faculty in computer science, and were familiar with graphical user interfaces and general data structures. None were familiar with the Simon Fraser fisheye view system.

*Materials.* The experiment was performed at the University of Calgary on Sun workstations, and used an implementation of the variable zoom algorithm developed in the Computer Graphics Laboratory at Simon Fraser University. The same software could be set to display either a full-zoom or a fisheye view of a simulated telephone network. The system allowed subjects to navigate through

the graphs and to change the status of links. Timing information and user events were automatically recorded.

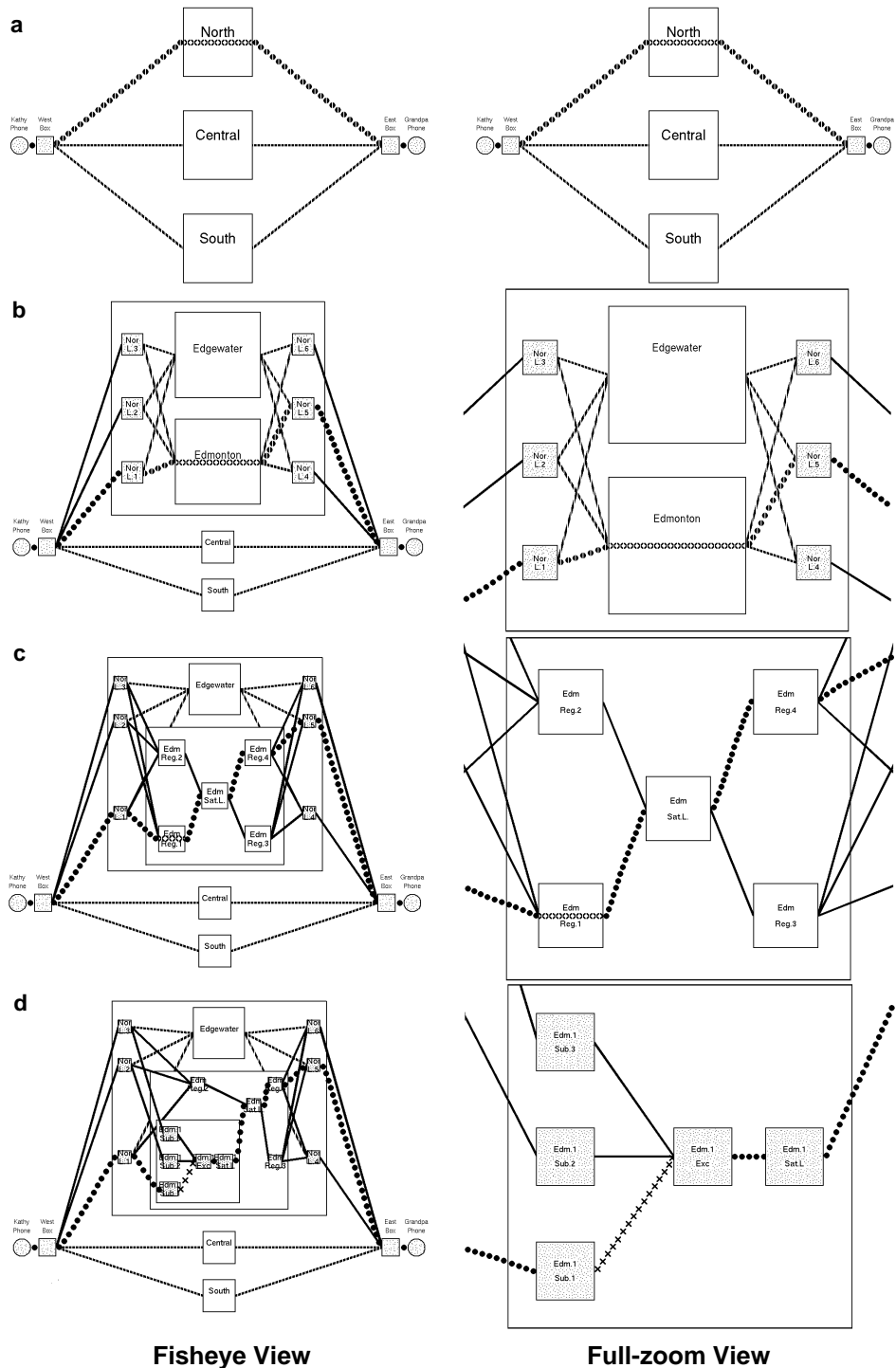
The fisheye view used the variable zoom method described in Section 2, with the balance factor,  $K_b$ , set to 0.5. The full zoom method followed a more traditional approach, where a selected node was enlarged to occupy the entire screen. From the users point of view, the only differences encountered during the experiment was in the visualization method. The system interface and the hierarchically clustered graph were otherwise identical.

Both systems illustrated a hierarchically clustered simulated telephone network with four levels of substations (nodes and clusters) represented by boxes and connected by lines (Figure 4). The entire network contained 154 nodes and 39 clusters. Each node was labeled to provide a reference to the node as well as contextual information. Figure 4a illustrates the root view of the network, comprised of four network nodes (the phones and east and west boxes), and three expandable clusters. The left column shows the fisheye view and the right the full-zoom view; in this case both views are identical. When a user clicks the left mouse button on the North cluster, that cluster is exploded to the next level of hierarchical detail. In this case, the fisheye view in the left column show the contextual detail around the exploded cluster, while the right shows only the cluster and its entry/exit links (but it is shown larger on the display). Similarly, expanding the “Edmonton” node and then the “Edm Reg 1” node will produce the displays seen in 4c and 4d.

Color was used to provide information about ‘telephone’ lines (the links), while texturing indicated user-selectability of nodes and lines. Only non-leaf nodes could be selected for zooming or unzooming, and lines could only be selected if both ends were connected to leaf nodes. Selection of a line resulted in its color changing. Rerouting of telephone lines was performed by selecting or deselecting connection lines i.e. by coloring or uncoloring the links between nodes (links between clusters were not selectable).

*Task Description.* Subjects were asked to act as telephone technicians. They were given a hierarchically clustered telephone network and asked to navigated through the network by zooming and unzooming nodes using the mouse. Subjects were first asked to find a broken telephone line in the network (seen as a red line, solid or textured depending on whether it connected leaf or non-leaf nodes) (Figure 4). After finding the break, subjects clicked on a button labeled “Break Found”. They were then asked to “repair” the network by rerouting a connection between two endpoints of the network that contained the break. Subjects then clicked on a button labeled “Reroute Done”, which ended the task.

*Methods.* Subjects were first given a short training task to perform, where they were expected to achieve some competency with the task and the software. Subjects then



**Fisheye View**

**Full-zoom View**

*These representations are adapted from the color screen image.*

●●●●●●●● represents the (green) colored, selected path.  
 - - - - - represents the (red) broken line.  
 Textured (speckled) nodes are not selectable.  
 Solid lines are selectable and dashed lines are not.

Figure 4. Snapshots at each level in the hierarchy for the fisheye (left column) and full-zoom (right column) views. Levels are: a) Root view, b) North cluster, c) Edmonton cluster, d) Edm Reg 1 cluster .

performed two similar navigational tasks, "Task A", followed by "Task B". Some of the subjects performed the first task, Task A, with the full-zoom view and the second task, Task B, with the fisheye view. The remaining subjects employed the same views in the reverse order. The training task was repeated using the second view before performing Task B.

The experimental design was a 2x2 factor ANOVA design, where the two factors (the independent variables) were the view and the order in which the views were performed (Table 1). Each factor has two levels, the view factor having levels "fisheye" and "full-zoom," and order having levels "fisheye-first" and "fisheye-second." The order factor should take into account both transfer effects and differences between the two tasks.

The dependent variables were the running time to complete the task, the number of zooms of network nodes that

		Order	
		Fisheye First	Fisheye Second
<b>View</b>	<i>Fisheye View</i>	S1-S9	S10-S20
<b>Factor</b>	<i>Full Zoom</i>	S1-S9	S10-S20

Table 1. The experimental design, showing the independent variables.

occurred during the task, and whether or not the task was successfully completed (if subjects did not correctly repair the break, the task was considered unsuccessful). Data collection was mostly automated. The software time-stamped and recorded every user event within the task (user selections, zooms and unzooms). However, the experimenters could only determine the existence of an error by retracing each subject's actions through the hierarchy.

Source	Sums of squares	Degrees of freedom	Mean squares	F-ratio	p
<i>a) Running Time</i>					
Order	1388.77	1	1388.77	.26	.616
View	31046.03	1	31046.03	9.91*	.006*
Order x View	10707.61	1	10707.61	3.42	.081
<i>b) Zooms</i>					
Order	17.6	1	17.6	1.13	.302
View	214.67	1	214.67	18.29*	.000*
Order x View	1.87	1	1.87	.16	.695
<i>c) Successful completion of task</i>					
Order	.26	1	.26	1.12	.303
View	.45	1	.45	2.32	.145
Order x View	.15	1	.15	.76	.395

Table 2. Anova Summary Table for all variables (\* indicates significant response)

View	Order	Mean	Std dev
<i>a) Running Time (seconds)</i>			
Fisheye	—	101.9	59.5
Full-zoom	—	161.2	71.6
Fisheye	First	113.5	72.9
Full-zoom	Second	136.6	62.9
Full-zoom	First	181.3	74.7
Fisheye	Second	92.4	47.6
<i>b) Number of Zooms</i>			
Fisheye	—	6.3	2.7
Full-zoom	—	10.9	4.3
<i>c) Successful completion of task as a ratio</i>			
—	—	.7	.46
Fisheye	—	.8	.41
Full-zoom	—	.6	.50

Table 3. Means and standard deviations of dependent variables at selected treatment levels

Qualitative comparisons between the two types of views were also gathered from subjects. We recorded their comments while they performed the experiment, and we administered a questionnaire after each task was completed. On each questionnaire subjects were asked to describe their strategy for solving the task, how they oriented themselves within the hierarchy, and what they liked and disliked about the system. After both tasks were completed, a final question asked each subject which view method they preferred using.

#### 4 RESULTS

We analyzed the running times of subjects to complete each task. An analysis of variance revealed that running time was significantly affected by the view factor used in the task (Table 2a), with people completing the task much faster when using fisheye views (102 seconds versus 161 seconds). While order did not have a statistically significant effect on the running times, the results hint that some learning may have occurred between the first and the second system tried; given the preliminary nature of this experiment, this should be revisited in future studies. The means and standard deviations of the running times are shown in Table 3a.

We also analyzed each subject's number of zooms on network nodes per task, which provides a quantitative measure of the amount of navigation required to complete the task. Note that "unzoom" actions were not analyzed, because nodes in the fisheye graph did not need to be unzoomed (since everything can remain visible on the screen). The analysis of variance revealed that the number of zooms was significantly affected by the view factor used (Table 2b), where subjects using the full zoom required almost double the number of zooms than in the fisheye views (11 versus 6 zooms). Differences due to ordering were not significant. The means and standard deviations of the number of zooms are shown in Table 3b.

Finally, we analyzed the number of correct solutions as a percentage of the whole. This was done by simply grading each task as correct (1) or incorrect (0); other than this, we did not attempting to assign a "value" of correctness for each solution. Neither view level nor order had a significant effect (Table 2c). However, the results weakly hints that view may influence performance; this should be reviewed in future experiments (Table 3). While most subjects did complete the task successfully, 30% of them did not. We believe that software enhancements, such as automatic checks for completion, would have improved all correctness values.

Some other effects beyond those analyzed statistically are worth noting. First, there was little difference in the performance of subjects when locating the broken telephone line within the hierarchy using either the full-zoom or fisheye system. This is because the display, independent of the views, clearly showed which of the lowest-level clusters visible on the display contained the

break. With a minimum of 4 operations required, the average of subjects using either view was 4.5 operations, with a standard deviation of 1.1.

Second, the ability of people to successfully complete a task deserves revisiting. No feedback on the condition of a path internal to a node was given, i.e. nodes only showed that paths entered into it but did not say if the paths were connected internally. Unconnected internal paths were usually the cause of an incomplete reroute. Eight of the twenty full-zoom reroutes attempted were not successful, while only four of the fisheye reroutes were incorrect. Incomplete paths were lacking connections and so the number of operations (zooms, unzooms, selections and deselections) was artificially low. While this apparently suggests that running time in the full-zoom case would be artificially lowered (since there were more tasks that were not completed), we did not find much difference in practice.

Third, people who successfully completed the task were able to produce far better reroutes through the network when using fisheye views. The raw data shows one-third of the correct reroutes using fisheye views were near the minimum possible number of operations; only a single subject had an extremely poor reroute. In contrast, no one using full-zooms came close to the optimum reroute; performance was generally poorer, and a full third of the solutions were extremely poor. There was a high degree of variance among subjects in the operations performed to complete a reroute. This is discussed further in section 5.1.

#### 5 DISCUSSION

This section interprets the results of the quantitative study, and discusses the qualitative responses obtained from subjects' comments and questionnaires. Our findings are then placed within the wider context of the network visualization research area.

##### 5.1 Examination of results

Subjects using fisheye views were more efficient at performing the task than when they used the full zoom technique. In particular, they took less time to complete the task, and the amount of navigation (indicated by the number of "zoom" actions) was reduced. This corresponded well to our subjective observations during the experiment—we saw that subjects using the fisheye view were able to focus directly on the task, and were not as distracted by the need to mentally visualize the network. In the questionnaires, most subjects also stated that they found the context provided by the fisheye view a valuable resource for completing the task.

The tasks consisted of two parts: finding the broken telephone line, and then rerouting the connection around that line. For both systems, subjects used the same strategy—a deterministic depth-first search—for finding the broken line. However, rerouting was performed using several different strategies. Most subjects attempted to use as much of the original connection path as possible, having

the reroute path be as close as possible to the original path (we call this local re-routing). This explains why people produced better reroutes in fisheye views, for the surrounding context easily showed them how nearby nodes were connected to each other. In contrast, subjects using full zoom had great difficulty doing local rerouting, for they became confused about their current position in the network, and they could not remember what nodes they had already examined. Several subjects, after attempting local rerouting, instead chose to reroute along a completely different path, starting from the top-down. While this reduced their confusion, it also required more selection of lines, which was very inefficient.

From their comments and responses to the questionnaires, most subjects greatly preferred the extra context provided by the fisheye view. It allowed them to concentrate directly on the task, and reduced their burden of trying to remember the structure of the entire network. But the choice is not cut and dried. Two of the subjects, for example, preferred the full-zoom system, and said that fisheye views presented a cluttered display that was difficult to work with. Two others qualified their preference of fisheye views by saying that their choice would depend on the task complexity and the size of the network. All subjects expressed more difficulty using the full-zoom view after using the fisheye view.

### 5.2 Limitations

Several problems and limitations became evident during the design and execution of this experiment. These included software limitations, design of a suitable telephone network, and problems with deciding how to represent "composite" edges in the graphs.

The software we used in the experiment was still under development; Simon Fraser University had created a special version to accommodate scheduling constraints at the University of Calgary. Thus, the software had several limitations that we expect to be repaired in future versions. The most noticeable distraction to subjects concerned the screen refresh; when subjects clicked on any edge to change its color, or on any node to expand it, the entire screen was redrawn. Only the affected area should have been refreshed. Better still, visual changes to the displayed network could be emphasized through animation, as done in Cone Trees or Perspective Wall (Mackinlay, Robertson et al 1991; Robertson, Mackinlay et al 1991). We believe this may alleviate a user's disorientation when zooming and unzooming nodes. Another minor distraction was the unnecessary accuracy required to operate the system: mouse clicks had to be exact, and accidental double-clicks occasionally resulted in errors.

Another design issue was the distinction between selectability and "composite-ness". Texturing indicated that a line or node was not selectable. However, this meant that both leaf nodes and "composite" lines (those with one or both ends at non-leaf nodes) were textured. Some confusion resulted when subjects correlated texturing not

with selectability but with whether the object was composite or not.

All these limitations were present in both the fisheye and full-zoom tasks. We would not expect their disappearance to change the results of the previous section.

### 5.3 Impact for practitioners

Our results have implications for designers of large information systems that are structured as networks. We suggest that the designer should consider if the information can be naturally represented to the user as a clustered hierarchy. If so, we believe that users are better able to manage the information space when the display provides both local detail and global context, as done through fisheye views.

There are surprisingly many real-world situations meeting these criteria. The particular telephone network and task used in our experiment is just one instance of the tasks generally found in many control rooms. Operators of real-time supervisory control and data acquisition (SCADA) systems often deal with hierarchically clustered networks such as power grids, machine plants, telephone systems, and gas pipelines. Operators must monitor the network operation. When something goes wrong with the network operation, alarms are sounded. Operators must then quickly isolate and repair problems; these are sometimes due to isolated failures of network components, or they could result from an interrelated breakdown of many components. Failure of these systems can affect large numbers of people, use expensive resources, and even be life-critical (e.g. a nuclear power plant operation). The operator must be able to navigate through these structures quickly and accurately.

Other situations where local detail and global context are important can be found in our daily computer usage. Much of the information we store in computers is hierarchical, such as computer file systems. Many graphical user interfaces to our file systems permit users to view several directories at once. However, these are often an all or nothing affair. Views do not show relations between directories (e.g. when links are allowed), and all information is shown at full size. Fisheye views could show these relations, and could give more visual emphasis to user's current items of interest.

### 5.4 Critical reflection

Our experimental results have shown that fisheye views provide a significant advantage over conventional full-zoom views. While the results are dramatic, some questions remain about the experiment and about fisheye views that should be addressed in further study; this would give us more certainty on how well we could generalize our results. Concerns include the choice of subjects, choice of tasks, training time, and the "degree" of fisheye view.

The subjects, drawn from an academic computer science environment, were familiar with both graphical interfaces and complex data structures (but not to control rooms).

While the mapping of task to an abstract network representation presented little difficulty to our subjects, other groups may find the mapping somewhat unnatural or confusing. It is possible that using a fisheye view—providing *more* information—may increase this confusion.

Training time in our experiment was minimal. While we do expect situations where users must be able to use a system with minimal training, there are also cases where significant training is the norm. There is, of course, a possibility that the user performance differences between fisheye and full-zoom views may be reduced (or increased!) after significant training time.

The two tasks used in the experiment, both involving maintenance of a telephone network represented as a hierarchical graph, may not generalize to other situations and domains. Still, we did ensure that the tasks involved navigation both from the root of the graph and also from deep within the graph. We believe this to be a typical problem, and suggest that a large class of tasks could benefit from fisheye techniques.

Not all fish-eye systems will give the same view. In this experiment, we used a particular style of fisheye views. However, there are many different fisheye views possible, by varying a *degree of interest* (Furnas 1986) function to emphasize or de-emphasize the full-zoom or fisheye quality of the display. Here, this was determined ahead of time by the experimenters on an arbitrary basis. Fisheye views present a tradeoff—global context vs. local detail. Also, the balance factor mentioned in Section 2 may be varied as well. While we believe fisheye views are superior to full-zooming, it is unclear exactly *which* fisheye view is appropriate for a given task. By incorporating different degree of interest functions and balance factors as a further variable in this experiment, an optimal tradeoff may be determined.

### 5.5 Research agenda

The data gathered here provides encouraging results towards the use of fisheye views for navigational tasks. Research, however, is far from complete. For example, how large can the network be before information overload becomes a problem, even using fisheye views? Will increased clutter undermine the benefits fisheye views provide? Is there an optimum number of hierarchical levels for a given network size and structure?

A few subjects expressed a preference for the simplicity of full-zoom views, because the amount of information presented on screen was always small. At the other extreme, the entire fisheye view network could be viewed on screen simultaneously (i.e., all clusters are expanded). The 154 nodes and 39 clusters in the simulated telephone network are near the limit of what could effectively be presented at once! Consider a cluster that has been expanded. Of the nodes and icons now visible, only a few may be truly useful to the task at hand. Perhaps *information filtering*, as an extension of Furnas' (1986) degree-of-

interest function, might make fisheye views more effective by pruning the “less useful” information from the display.

Multiple foci (independently zoomed nodes) were allowed by the software used in this experiment, though their use was not examined. The potential benefits and/or problems of multiple foci are unclear and need to be examined.

In terms of the tasks being tested, improvements to the system might include an unobtrusive “spontaneous interest” indicator. The motivation for this comes again from the human eye, where motion—even far from the focal point—is a key determiner of interest. A gentle but persistent motion (e.g. vibration) might be used to indicate a problem that may otherwise be hidden in a reduced path or node. Motion would tend to stand out well in an otherwise static display, and would highlight trouble spots needing attention.

Another research direction is how fisheye views can be extended to other data structures. We dealt here only with a hierarchically clustered 2-d network.

## 6 CONCLUSIONS

We proposed the variable zoom algorithm for generating fisheye views of hierarchically clustered networks. We then described an experiment contrasting fisheye views with traditional full zoom views. Results suggest that the greater context provided by fisheye views significantly improved a user's performance of the tasks. Using the fisheye, subjects were able to concentrate directly on the task itself, resulting in quicker navigation and less unnecessary exploration. We suggest that fisheye viewing interfaces should be favored over traditional viewing approaches for displaying large information spaces. Further work remains to determine how the significant advantages of fisheye views reported here will generalize across different levels of task complexity and to other data structures.

## ACKNOWLEDGMENTS

Development of the variable zoom method was part of the Intelligent Graphic Interface project, made possible through the support of Industry, Science and Technology Canada, of the British Columbia Ministry of Advanced Education, Training and Technology and of PRECARN Associates. Research at the University of Calgary was partially funded by the National Science and Engineering Research Council.

The graduate course of Human Computer Interaction at the University of Calgary provided feedback to the experiment. Subjects freely gave their time and energies; they deserve a special thanks. Troy Brooks of Simon Fraser's Computer Graphics Lab developed much of the data logging software.

## REFERENCES

- Beard, D. V. and Walker, J. Q. (1990) “Navigational techniques to improve the display of large two-dimensional spaces.” *Behaviour and Information Technology*, 9(6), pp. 451-466.
- Berge, C. (1973) *Graphs and Hypergraphs*, North-Holland,

- Amsterdam.
- Fairchild, K. M., Poltrock, S. E. and Furnas, G. W. (1988) "SemNet: Three-dimensional graphic representations of large knowledge bases." *In Cognitive Science and its Application for Human-Computer Interface*, pp. 201-233, R. Guindon ed. Elsevier.
- Furnas, G. W. (1986) "Generalized fisheye views." *In Proc ACM CHI'86 Conference on Human Factors in Computing Systems*, pp. 16-23, Boston, Massachusetts, April 13-17, ACM Press.
- Harel, D. (1988) "On visual formalisms." *Communications of the ACM*, **31**(5), pp. 514-530.
- Johnson, B. and Shneiderman, B. (1991) "Tree-maps: A space-filling approach to the visualization of hierarchical information structures." *In Proc IEEE Visualization '91*.
- Mackinlay, J. D., Robertson, G. G. and Card, S. K. (1991) "The perspective wall: Detail and context smoothly integrated." *In Proc ACM CHI'91 Conference on Human Factors in Computing Systems*, pp. 173-179, New Orleans, Louisiana, April 28-May 2, ACM Press.
- Robertson, G. G., Mackinlay, J. D. and Card, S. K. (1991) "Cone trees: animated 3d visualizations of hierarchical information." *In Proc ACM CHI'91 Conference on Human Factors in Computing Systems*, pp. 189-194, New Orleans, Louisiana, April 28-May 2, ACM Press.
- Sarkar, M. and Brown, M. H. (1992) "Graphical fisheye views of graphs." *In Proc ACM CHI'92 Conference on Human Factors in Computing Systems*, pp. 83-91, Monterey, California, May 3-7, ACM Press.
- Schaffer, D. and Greenberg, S. (1993) "Sifting through hierarchical information." *ACM INTERCHI'93 Conference on Human Factors in Computing Systems*. Amsterdam, April 24-29. Poster presentation.
- Shneiderman, B. (1991) "Tree-maps." *In IEEE Visualization*. Excerpts from the Video.