Multi-Fisheye for Interactive Visualization of Large Graphs

Priya Sundararajan, Ole J. Mengshoel, Ted Selker
Carnegie Mellon University, Silicon Valley
{priya.sundararajan, ole.mengshoel, ted.selker}@sv.cmu.edu

Abstract
By selectively zooming in and zooming out visualizations, the fisheye technique allows users to study details while maintaining context. In this paper, we introduce a multi-fisheye technique, which amounts to introducing several fisheyes in a visualization at the same time. Our multi-fisheye technique is based on partitioning the visualization’s display area and applying a fisheye algorithm inside each partition. While we demonstrate the potential of applying our multi-fisheye technique using a social network, it clearly can be applied in other areas and types of networks, including in probabilistic graphical models such as Bayesian networks.

Introduction
Interactively exploring large data sets, such as Bayesian networks (Pearl 1988) with hundreds or thousands of nodes and edges, can be quite challenging. In particular, users may lose context when studying and understanding zoomed-in details. Also, browsing a large layout by scrolling and arc traversing tends to obscure the global structure of the graph (Henry and Hudson 1991).

To address the fundamental visualization problem of studying details while also giving context, the fisheye technique was introduced (Furnas 1986). The key idea with fisheye is that the user selectively applies a distortion, such that the center of the fisheye distortion becomes zoomed-in, while the parts of the display further from the center become zoomed-out. Fisheye lenses have been elegantly applied, for instance, in graphical layouts (Gutwin and Fedak 2004) and menu layouts (Hornbaek and Hertzum 2007).

While very useful, the fisheye distortion allows focus on only one part of the display, for example one part of a network. In many cases, however, a user wants to compare multiple different areas of a visualization, areas that in general can be quite far apart. Such comparison can be achieved by sequentially applying the fisheye distortion and remembering previously zoomed-in areas when studying the currently zoomed-in area of the display; however this approach becomes somewhat tedious.

We explore a new way of using fisheye, a multi-fisheye technique, and apply it to the problem of visualizing and interacting with large graphs. To help users compare multiple parts of a graph while minimizing distortions, we decided to localize the fisheye effect to a region surrounding a selected node. This is done by applying the fisheye algorithm (Sarkar and Brown 1992) locally to each of the selected node and its sphere of influence. Thus many fisheyes are created at the same time. The neighboring nodes around the fisheye is selected by partitioning the display. We briefly discuss several partitioning algorithms, including the Voronoi algorithm (Fortune 1986). Based on a set of selected nodes (each representing a fisheye), the Voronoi algorithm divides the plane into polygonal regions. Each region contains a set of nodes that are closer to a particular selected node than to all other selected nodes. Fisheye is then applied to those nodes in the selected region, allowing users to zoom a portion of the graph without losing context.

Related Work
Typical graph presentation might use all display space for a particular zoomed-in area of a graph. This provides focus, but at the expense of losing overview of the entire graph. Sometimes the portion of the graph will be identified in a separate overview with the zoomed area delimited as well, now a well known approach it might have first been implemented by Ted Selker and Scott Penberthy as a mechanism for selecting personnel in zooming the management graph of IBM employees in 1986. Another popular way to retain overview and detail is the fisheye approach (Furnas 1986). In a fisheye view, the user-selected portion expands, or is zoomed-in, while the nearby regions contract, or are zoomed-out, allowing an entire network to be viewable while the region of interest is expanded for focus. Hyperbolic viewers (Lamping and Rao 1994) are a well-known related variant. van Ham and Perer developed the “search, show context, expand on demand” model in which the users pick a particular datapoint as a focus for analysis; the system then computes and displays an “optimal” relevant context given the user’s current interests (van Ham and Perer 2009). They used the concept of Degree Of Interest (DOI) (Furnas 1986; 2006) to display the relevant context. A topological fisheye method (Gansner, Koren, and North 2005) precomputes coarsened graphs on the fly and render the level of detail from the combined graphs, depending on distance from one or more foci. This topological approach has not been celebrated as we think it should. The drawback of this topo-
logical approach, however, is the computation involved in precomputing the coarsened graphs.

A study conducted by Plumlee and Vare compared systems in which multiple partial views were compared to a single zoomable interface. For a multiscale pattern matching task (Plumlee and Ware 2006) shows that the multiple view interface seems to work better than a zooming user interface. The disadvantage is the extra screen space used by the multiple view interface. The users will also find it difficult to manage when more views are opened.

There are many social network visualization tools. SocialAction provides attribute rankings and coordinated views to zoom, filter clusters of nodes and isolate outliers (Perer and Shneiderman 2006). NodeXL provides the Excel users to easily import, clean up, analyze and visualize data using a library of standard visualization techniques (Smith et al. 2009). ManyNets is another system that uses a tabular visualization to display multiple networks (Freire et al. 2010). TreePlus allows users to explore graphs presenting them as a familiar tree layouts (Lee et al. 2006). None of these techniques and systems have mechanisms for browsing for nodes in different parts of the network for the purpose of comparing easily. Our multi-fisheye system described below, on the other hand, supports interactive segmentation for comparing multiple parts of a graph.

**Approach**

Fisheye technique is independent of the layout algorithm and is defined as a separate processing step on the graphical layout of the graph. This independence has positive and negative aspects. On the positive side, it allows for a modular organization of software (Herman, Melancon, and Marshall 2000). However, the fisheye distortion may destroy the ability of the user to recognize the already established relations in the graph by distorting the display. To combat the negative aspect of fisheye, we decided to localize the fisheye effect on the bounded area around the focus. We tried two different approaches, the rectangular partitioning technique and the Voronoi partitioning technique. These two techniques are discussed below.

**Rectangular Partitioning Technique**

In this approach we divide the screen space into rectangles and apply fisheye only in the selected area. We show an example of partitioning the screen space into four fixed quadrants in Figure 1. When a node is selected, its quadrant is determined and all the nodes in that quadrant undergo fisheye distortion. When multiple nodes are selected, each of them will be zoomed using the fisheye technique.

**Voronoi Partitioning Technique**

The constraint in the rectangular partitioning technique is that all the nodes inside a rectangle will undergo fisheye distortion. To optimize the fisheye distortion process, the number of nodes undergoing distortion need to be reduced. For this a recursive partitioning technique is needed for creating new partitions at each node selection. This is done by applying the Voronoi algorithm and then doing fisheye distortion in the newly created partitions. The Voronoi diagram can be maintained for a set of partitions that varies over time by insertion or deletion (Aurenhammer 1991) as shown in Figure 2.

The Voronoi diagram of the display area is computed by dividing the plane into \( n \) selected regions for each of the \( n \) node selections. It is based on the Voronoi principle that any node in the region will be nearer to the selected node in that region than to any of the neighboring selected nodes. Thus the diagram forms zones of interest for the selected node, which helps to pick only those nodes in the zone to apply fisheye technique.

When the user selects a point, the Voronoi algorithm is applied to that point along with all previously selected (and not de-selected) points in order to compute the Voronoi diagram. A divide and conquer approach is followed wherein the screen space is divided into polygons and the intersecting edges are merged to form a bounded region. We apply the fisheye only to the bounded area by retrieving the corner co-ordinates of the bounded region and updating the display accordingly. In this way, the distortion of the entire graph is minimized and the layout structure is retained.

The pseudocode for the above methods is shown in Figure 3. After the original graph is rendered, we wait for the user to select a node. When the user selects a node, the MultiFisheye algorithm is called. The display space is partitioned depending on whether the user wants to partition using the rectangular or the Voronoi algorithm. The corner coordinates (endPoints) of the selected nodes’ partition is obtained. Each node in the graph is checked to see if it is present in the selected nodes’ partitioned area using the Ray casting algorithm 1. Only those nodes in the selected nodes’ partitioned area undergoes the fisheye distortion.

**Implementation and Interaction**

The multi-fisheye technique is implemented using the Prefuse framework (Heer, Card, and Landay 2005). A Social

---

1http://www.ecse.rpi.edu/Homepages/wrf/Research/Short'Notes/pnpoly.html
network is taken as an example graph structure. The plain graph with no fisheye distortion is shown in the left side of Figure 4. The font size of the labels is the same for all the nodes. Interaction with the graph takes place by changing the position of focus. The zoomed node will display a larger font than its neighboring nodes. Detailed description of each of the interaction mechanisms are discussed in the following sections.

Traditional Single Fisheye Technique

If the entire graph is a single rectangle, the system lays out the graph as a single fisheye system would. As a user selects a node, it gets enlarged, all other nodes sizes are reduced by the log of their distance from the selected node (Sarkar and Brown 1992).

Novel Multi-Fisheye Technique

We have investigated two partitioning approaches, the rectangular approach and the Voronoi approach. In the rectangular approach, if the user partitions the graph into more than one rectangle, the nodes in each rectangle will be laid out as though they were separate graphs. As a user selects a node within a rectangle, it will be laid out with constraints of continuity at the edges to attach to the other rectangles. Multi-node selection will mainly be used for comparing various nodes to explore their differences and similarities. In the rectangular algorithm, the partition is fixed. When multiple nodes are selected within a rectangle then each of the selected node and the surrounding nodes will undergo fisheye as shown in Figure 1. The right side of Figure 4 shows the rectangular partitioning technique applied to a selection of three nodes.

Another way to partition the graph relies on a Voronoi algorithm (Fortune 1986) to partition the display area. While such partitioning might seem more general, it might be harder to keep track of where the lines are between the regions that the fisheye applies to. With either Voronoi or our implemented rectangular segmentation approaches, only the nodes inside the partition will undergo distortion. Thus these approaches allow many foci to raise specific nodes to be compared with other expanded nodes in other partitions. Using the Voronoi approach, we can also easily undo the effect of one or more of the fisheyes.

Conclusion And Future Work

This paper introduces multi-fisheye as an approach to partitioning large scale graph visualizations. The multi-fisheye approach attempted in (Gansner, Koren, and North 2005) is simplified and developed for the 2D graph case. For multi-zooming, a set of focus points are selected and those can be zoomed using the localized fisheye algorithm, at the same time retaining the context of the graph. Thus the approach allows a user to bring multiple parts of a large graph for-
Algorithm

: **PARTITION AND FISHEYE** (Graph)

```plaintext
procedure VORONOI(selectedNode)
  comment: Partition display around the selectedNode
  endPoints ← FORTUNES VORONOI (selectedNode)
Voronoi Algorithm (Fortune 1986)
return (endPoints)

procedure RECTANGULAR(selectedNode)
  comment: Partition the display area into 4 quadrants
  Get selectedNode's position
  nodePoint ← nx, ny
  Check which quadrant contains the nodePoint
  Get the endPoints of the quadrant
return (endPoints)

procedure FISHEYE(Graph, selectedNode, endPoints)
  comment: Check if the node is in the partition
  comment: Apply the fisheye technique only to surroundNodes
  for each Node ∈ Graph
    if CHECK COORDS(Node, endPoints) == TRUE
      surroundNodes ← Node
  FISHEYE ALGORITHM (selectedNode, surroundNodes)
  Fisheye Algorithm (Sarkar and Brown 1992)

procedure MULTI FISHEYE(Graph, selectedNode, method)
  comment: Call the partitioning method
  if method == ‘Rectangular’
    endPoints ← RECTANGULAR (selectedNode)
  else endPoints ← VORONOI (selectedNode)
  FISHEYE(Graph, selectedNode, endPoints)

main
  method ← partitionMethodName
  comment: method denotes the partitioning technique
  Graph ← Render the original graph
  When the user clicks on a node, call MultiFisheye
  MULTI FISHEYE(Graph, selectedNode, method)
  for each Node ∈ Graph
    POSITION BACK NODE()
    Redraw the Nodes to its original position
```

Figure 3: Pseudocode for Fisheye technique applied on spatial partition.

ward to be compared in localized fisheye distortions. Partitioning the graph using the Voronoi algorithm was considered. We also developed a new rectangular graph segmentation approach that is a simpler algorithm. The interactive multi-fisheye viewer described in this paper has been used to help view large Bayesian networks. The system focuses on the ability to keep track of the relationships in a large network and compare various parts of a large network. It is also conceivable that 3D visualizations could, with care, benefit from such multi-fisheye techniques. If combined with a multi-view technique (Cossalter, Mengshoel, and Selker 2011), we anticipate that multi-fisheye could be part of an integrated tool for editing and analyzing Bayesian and other analytical networks.

References

Herman, I.; Melancon, G.; and Marshall, M. S. 2000. Graph visualization and navigation in information visualization: A


