A Space of Presentation Emphasis Techniques for Visualizing Graphs

Emanuel G. Noik
Computer Systems Research Institute
University of Toronto
6 King's College Road
Toronto, Ontario, Canada m4s 1a1

e-mail: noik@db.toronto.edu
Telephone: (416) 978 8609

Abstract

The graph topo-visual formalism has been shown to be well-suited to the task of visualizing complex relations on a set of elements. Unfortunately, most visual formalisms do not scale very well. This observation is particularly true of graphs, which even when hand-drawn by an artist, are seldom meaningful when the number of nodes or links exceeds a very modest threshold – typically only a few hundred elements. This severe limitation has prompted many researchers to seek alternative visualization techniques that may eliminate, or, at the very least, raise this threshold.

In this paper we analyze these recent efforts, describe an abstract space of presentation emphasis techniques, and locate the current approaches within this space. The contributions of this paper are several: (1) a significant portion of recent work is collected and reviewed; (2) a common set of criteria and a taxonomy of graph views are proposed; these, (3) permit a more direct comparison of previous work; which helps to, (4) identify common shortcomings and limitations; which in turn, (5) suggest future directions.

Keywords: presentation emphasis techniques, fisheye views, relational data visualization, graphs, nested graphs.

1 Introduction

A perusal of recent literature in relational data visualization and allied fields reveals a smattering of references to a variety of presentation emphasis techniques intended to improve the visualization of large information spaces. To a large extent, these techniques have been aimed at the visualization of a variety of graph structures: hierarchies, directed graphs and networks, general undirected graphs, and nested (hierarchically clustered) graphs. The graph has long been recognized as an effective topo-visual formalism; whereas a graph depicts binary relations on a set of elements, a nested graph can, in addition, express one-to-many relations [8].

Although the number of papers that deal directly with emphasis in graphical presentation is relatively small, the vocabulary used to describe fundamental concepts has grown somewhat large and inconsistent. More distressing, however, is the fact that although a variety of sophisticated emphasis techniques have been described in the literature, there is still no common framework for comparing these techniques, measuring their "expressive power", and validating their overall effectiveness. While we do not suggest that we have solved these difficult problems, we hope that we have nevertheless taken a first step to remedy the situation.

This paper is organized as follows: Section 2 defines basic concepts, terminology, and a taxonomy of graph views; Section 3 summarizes and compares recent presentation emphasis techniques in relational data visualization; Section 4 describes an abstract space of emphasis techniques and locates the individual techniques within this space; and Section 5 identifies common limitations and suggests future research goals.

2 Terminology

2.1 Graphs and Graph Diagrams

A *graph* $G = \langle V, A \rangle$ consists of a set of *vertices* V, and a set of *arcs* $A \subseteq V \times V$. A *nested graph* $N = \langle V, A, C \rangle$, also contains a set of *containment arcs* $C \subseteq V \times 2^V$; in most cases, the containment relation is restricted to be hierarchical (*i.e.,* C is a forest of trees). Vertices, arcs, and containment arcs may be labelled.

A graph *diagram* or *layout* is a visualization of the graph in which vertices are typically represented by *nodes*, arcs are depicted as *links*, and containment arcs are usually displayed as *boxes* (closed rectangular or polygonal regions) that fully enclose the nodes associated with the nested vertices. Graph layouts may be hand-drawn or may be generated automatically by a *graph layout algorithm* [4]. In addition to textual labels, nodes may also have additional content (*e.g.,* graphical icons).

2.2 Emphasis-related Concepts

The following terminology comes in large part from the generalized fisheye view (*FEV*) formalism [7].

The generalized *FEV* metaphor is based on the workings of the fisheye or very wide angle lens used in photography, which magnifies the image at multiple levels – greatest near the lens' focus and least in its periphery. By balancing local detail and global context, *FEV*s can simultaneously display information at multiple levels of abstraction.

When displaying large structures, the basic strategy uses a *degree of interest* (*DOI*) function to assign to each point in the structure a number, or *priority*, that quantifies the user's interest in that point given the current task. Priorities may be assigned by the user, or may be automatically computed by a *priority algorithm* (*PA*). In a generalized *FEV*, the *DOI* is decomposed into two components: *a priori importance* (*API*) which computes the global importance of any point in the structure, and *distance* (*Dist*) which computes the conceptual distance between any two points. If one point is selected as the current focus of interest, or *focal point* (*FP*), then in its simplest additive form, *DOI*(p) = *API*(p) - *Dist*(p, f) is the user's degree of interest in point p given *FP* f; thus *DOI* increases with *API* and decreases with *Dist*.

The *proximity* (*Prox*) is typically defined as the difference of the maximum distance and a given distance (*i.e.,* *Prox*(u, v) = *Dist*$_{max}$ - *Dist*(u, v)).

An *emphasis algorithm* (*EA*) is defined in terms of a *PA*, the graph view (see the next Section), and the mapping between the priority values and the associated presentation variables (*e.g.,* size, colour, line style and thickness). In *EA*s for graphs, a *FP* is typically a vertex or a node, but can also be a coordinate in the 2-D or 3-D layout; unlike the generalized *FEV*, some *EA*s are defined in terms of multiple *FP*s, and may use different *DOI* functions to compute priorities.

2.3 A Taxonomy of Graph Views

A *normal view* is a layout in which all elements have identical priorities, or, equivalently, a view with no *FP*s. An *implicit FEV* arises from the effect of point perspective in 3-D, by which nearby points loom large and distant points appear small; implicit *FEV*s are usually static (*e.g.,* [5, 14, 22]), but may be dynamic if the 3-D layout is permitted to be altered (*e.g.,* by moving higher priority nodes to the foreground [21]). There are at least three types of *emphasized views* (*EV*s). A *filtered view* displays a subset of elements and suppresses the rest – this can be accomplished by zooming or by filtering according to elements' priorities (display elements with priorities above some threshold, suppress others). A *distorted view* emphasizes elements by distorting their sizes, shapes, and positions – either by a polar or orthogonal distorting *FEV* transformation (*Dist* is geometric), by a non-geometric *FEV* distortion (*Dist* is non-geometric), or by scaling at single (local) or multiple (global) levels. An *adorned view* emphasizes elements by varying other visual presentation variables such as colour, shading, line style and thickness, as well as audio [13], and motion (*e.g.,* "in-betweening" animation [21], vertical oscillations and small random movements [5], vibration or pulsing [27]). Figure 1 shows a normal view of this taxonomy, while Figure 2 shows examples of other views of the same taxonomy. Note that many of the *EA*s described in the next Section generate hybrid views that exhibit the properties of two or more of these basic views.

3 A Review of Recent Work

We've divided our review into two sections: approaches that are not specifically geared towards the visualization of graphs and those that are.

3.1 Non-graph-oriented Techniques

Generalized Fisheye Views. This formalism for reducing the complexity of highly detailed visualizations by displaying subsets of the most relevant details [7], generates filtering *FEV*s. As mentioned earlier, in a generalized *FEV*, *DOI*(p) = *API*(p) - *Dist*(p, f), given *FP* f.

Aircraft Maintenance Diagrams. This *EA*, an extension of the generalized *FEV* strategy for presenting aircraft maintenance data [18], generates filtering *FEV*s. The overall goal of this work was to enhance the usefulness and effectiveness of computer-based maintenance and diagnostic tools for aircraft repair technicians. Although the visualizations in this domain were circuit schematics and mechanical parts diagrams, the underlying data model was a directed network in which vertices represented components and arcs represented physical and functional relationships between the components. The generalized *FEV* formalism was extended to capture multiple *FP*s in a directed network by rewriting *Dist* as the sum of shortest path distances from a vertex to the *FP*s. The resulting prioritization of vertices enabled the construction of nth order *FEV*s, similar to the generalized *FEV*s of trees [7].

The Perspective Wall. This *EA* for visualizing linear information [15] such as time, generates globally scaled views. It uses graphics hardware support to fold wide

2-D charts into 3-D visualizations consisting of a center panel for showing detail and two perspective panels for displaying context. The perspective view provides efficient space utilization for 2-D charts with wide aspect ratios, and generates a *FEV* effect: it emphasizes the neighbourhood of the detailed view by making it larger than the more distant parts of the contextual view. The trade-off between detail and context is controlled by manipulating the degree of folding, the width of the detail panel, and the angle of the field of view. The single *FP* is changed by scrolling the wall as one would a sheet in a player piano.

Tree-Maps. This *EA* for presenting hierarchical information [12, 13, 28, 32], generates hybrid locally scaled and adorned views. A hierarchy is drawn as a set of nested boxes in which each node is depicted as a rectangular region composed of the rectangular regions that represent its children. The main advantage of the method is its ability to visualize large hierarchies, which it achieves through its linear top-down space-filling layout algorithm. Tree-Maps can emphasize nodes in the hierarchy by a two step process. First, each node is assigned a weight (by the user of application) subject to the constraint that this weight is greater than or equal to the sum of the weights of the node's children. Next, the layout algorithm ensures that the total area in the drawing allocated for each node is proportional to its weight. In this manner, more important nodes are emphasized by being drawn larger than nodes of lesser importance.

Notice that while Tree-Maps provides the mechanism to generate visualizations that incorporate emphasis, it does not solve the problem of assigning meaningful weights to nodes given a set of *FP*s.

StretchTools. This *EA* [25] used to manipulate 2-D screen space, generates locally scaled views. The user interface is based on the metaphor of stretching a rubber sheet by using *handles* and *clamps*. A handle, when placed on the screen and pulled causes the screen (and the graphical objects displayed on the screen) to expand on one side of the handle and contract on the other. Clamps impose constraints on the movement of handles by joining handles with one another.

Although the *EA* demonstrates the potential effectiveness of the rubber sheet metaphor, the approach has several undesirable properties. First, the technique does not provide a way to enlarge a single region without affecting an entire row or column of regions. Second, because local scaling is used, the regions near a *FP* are not enlarged automatically, but remain as small as remote regions; to generate a *FEV* effect, more handles must be explicitly added to scale diagrams at multiple levels. Third, orthogonal stretching caused large discontinuities at the region boundaries failing to smoothly integrate individual regions into a coherent layout.

On the positive side, since the *EA* can be used to stretch a screen that contains arbitrary graphical objects, it follows that the approach can be used to manipulate drawings of graphs – both flat and nested.[1]

Morphing. This *EA* [26] is based on image transformation technology and generates globally scaled views. It was proposed to overcome the difficulties associated with StretchTools [25], and allows users to specify polygonal *FP* regions; global scaling is used to enlarge the interior of the selected regions while simultaneously adjusting the remainder of the layout so that the magnified areas are smoothly integrated with the demagnified ones.

3.2 Graph-oriented Techniques

Generalized *FEV*s of Trees. This *EA* for simplifying the display of trees by suppressing less relevant nodes [7], generates filtering *FEV*s. A generalized *FEV* of a tree is obtained by providing specific instantiations of the *API* and *Dist* functions. Specifically, *Dist* is the path distance between two vertices, *API* is the distance from the root of tree, and $DOI(v)$ = - $(Dist(v, f) + Dist(v, root))$. This treats the internal nodes as intrinsically more important than the leaves of the tree. An nth order *FEV* is obtained by only displaying points with $DOI(v) \geq -(3 + 2 * n)$.

SemNet. Three *EA*s for the 3-D networks of SemNet were proposed: clustering (filtering *FEV*), 3-D point perspective (static implicit *FEV*), and sampling density (filtering *FEV*) [5]. In clustering, recursive subdivision of the 3-D graph volume yielded an 8-ary tree with SemNet nodes partitioned among its leaves; tree distance (an approximation of Euclidean distance) was used in the *DOI* function. Although not implemented, the third *EA* was motivated by the workings of the human retina which samples an image densely at its center of focus and in successively less detail for points further away. All three *EA*s were defined in terms of a single *FP*.

Topographic Networks. This *EA* for displaying topographic networks [10, 11], generates hybrid filtering fisheye and non-geometric distorted fisheye views. The *FEV* was formulated in terms of a single *FP*; the *DOI* metric was used to both, determine the amount of detail to be displayed (*e.g.,* omit node labels), and to distort the positions and sizes of nodes.

[1] In fact, the authors show an example of stretching screens generated by the prototype graph browser which was used to display graphical *FEV*s [23, 24].

Cone Trees. Cone trees display hierarchical information by using 3-D graphics and interactive animation [22]. By limiting the domain of inputs to hierarchies, Cone Tree visualizations were not exposed to many of the problems that complicated the design of SemNet [5] and increased the cognitive load placed on its users. Cone tree visualizations support two types of *FEV*s: distortion from 3-D point perspective (static implicit *FEV*), and filtering through interactive *gardening* operations (filtered view). Additionally, the Cone Tree search facility automatically rotates cones to bring the target node to the front of the display, thereby relieving the user from an otherwise difficult manual searching task. Gardening consists of three types of operations: in *pruning*, the descendants of a selected node are hidden; *growing* restores hidden nodes; the *prune others* operation prunes the siblings of the selected node leaving only the selected substructure visible. The last operation, in particular, allows a user to interactively select *FP*s, and can be used in conjunction with the search facility to simplify a large, complex hierarchy. Ironically, the highly interactive nature of the technique makes it less appropriate for exploring very large hierarchies – a more automatic solution would be preferred (*i.e.,* it is unrealistic to expect users to work hard to simplify visualizations of large hierarchies).

Compound Digraph Display Methods. Three multi-viewpoint perspective (MVP) *EA*s [17, 29] have been applied to drawings of compound digraphs [30]: the fisheye *EA* uses a polar transformation to map each point in the original drawing to the perimeter of a circle (polar distorting *FEV*); the orthogonal fisheye *EA* uses independent Cartesian transformations to map each point to the perimeter of a square (orthogonal distorting *FEV*) meaning that lines that were parallel to the x or y axis in the normal view remain parallel to these axes in the distorted view; in the biform *EA*, *FP* areas are magnified uniformly and others are de-magnified uniformly (locally scaled view).

A related publication [3] describes three properties that should be preserved when a drawing is transformed:

1. *Orthogonal ordering:* preserve the vertical and horizontal ordering of points (*e.g.,* if p is north-east of q in the normal view then p' must be north-east of q' in the distorted view).

2. *Topology:* the inside of a closed continuous curve must be mapped to the inside of a closed continuous curve (*i.e.,* if there is no overlap in the normal view, there must be no overlap in the distorted view).

3. *Clusters:* objects that are proximate in the normal view must also be proximate in the distorted view.

Document Associative Networks. This *EA* for simplifying the task of information retrieval [6], generates hybrid filtering fisheye, globally scaled, and adorned views. The approach is centered around a common visually displayed network structure: an associative network. The user interface visualizes networks using the generalized *FEV* formalism in which a node's *API* is its degree (number of incident arcs) and *Dist* is the sum of arc weights along a shortest path. A node is displayed only if its *DOI* exceeds a threshold. Inexplicably, however, only the *API* rather than the *DOI* is used to determine the size of each displayed node. In addition, shading is also used to further emphasize interesting (larger) nodes. The *FP* (node) is drawn as large as the largest node in the *EV*.

The shape of links is also used to convey information about the structure of the network: links are widest at the *FP* and narrow as they connect nodes that are more distant from the *FP*.

Interactive Graph Layout. This approach comprises interactive or algorithmic subgraph selection, layout algorithm assignment, and sublayout composition [9], and generates hybrid locally scaled and adorned views. The nodes in a selected subgraph can be enlarged to emphasize the subgraph. For example, to emphasize a path, the nodes on the path can be positioned by a row or column layout algorithm, their bounding boxes can be magnified, and an alternative font (type and size) can be used to render their textual labels. Since interactive layout algorithm assignment is required to specify multiple local scaling distortions, it is unclear whether the technique is appropriate for generating *EV*s of large graphs.

Graphical FEVs of Graphs. This technique [23, 24] generates hybrid filtering fisheye and distorting fisheye (polar and orthogonal) views. "Graphs" implies that the technique can be applied to non-hierarchical graphs, while "graphical" implies that the technique furnishes a graphical interpretation of *FEV*s which integrates layout considerations into the fisheye formalism: the position, size, and level of detail of displayed nodes are computed based on client *PA*s.

A graphical *FEV* is obtained by magnifying the nodes of greater interest and correspondingly demagnifying nodes of lesser importance, and recomputing the positions of all nodes and link bend points. Although the client *PA*s can be modified to ignore the geometry of the normal view, the technique has been designed to function primarily as a distortion of existing layouts. For example, the position of a node in the *FEV* is a function of its position and the position of the *FP* in the normal view (*i.e., Dist* is the Euclidean distance between nodes in the normal view).

Although graphical *FEV*s can be implemented efficiently using dedicated graphics hardware, the technique is limited to non-nested graphs with a single *FP*.

Abridgment. This *EA*, used in the D-ABDUCTOR system [16] for viewing and manipulating compound digraphs [30], generates hybrid filtering fisheye and non-geometric distorting fisheye views. Priorities are computed as a linear combination of three quantities: structural importance (vertex nesting depth); semantic importance (*API*); and focal importance (proximity to the set of *FP*s). Priorities can be used in two ways: in a hybrid drawing, nodes with priorities less than a lower threshold are suppressed, those between the lower and upper thresholds are reduced, while those with priorities greater than the upper threshold retain their original sizes; in a proportional drawing, nodes' sizes are directly proportional to their priorities. Animation can be used to reduce abrupt changes caused by a change in the set of *FP*s.

Variable Zoom. This *EA* [27] generates hybrid filtering fisheye and globally scaled views of hierarchically nested graphs. The *EA* is similar to the Biform Display Method [17], and is limited to drawings in which the projections of nodes on the x and y axes do not overlap. The contents of each node are recursively rescaled in a top-down fashion. Within each node, selected nodes are magnified while the rest are demagnified; the *EA* computes scale factors that preserve a node's size while resizing and repositioning its contained nodes. A balance factor is used to control the ratio of detail to context at each level in the hierarchy, meaning that all zoomed nodes at a given level are magnified equally. Variable Zoom, like StretchTools [25], was designed for interactive rather than automatic operation – the user interacts to selectively magnify and demagnify portions of the display. Unlike StretchTools, however, which permits arbitrary distortions of the screen, the distortions obtained by this *EA* are more restrictive due to its use of the balance factor as described above. Furthermore, without explicit notions of *DOI* and decomposition into *API* and *Dist*, these and other techniques [9, 15, 17, 22] are less appropriate for automatic generation of *EV*s.

Fractal Views of Trees. This *EA* exploits fractal self-similarity to aid the visualization of large trees in 3-D [14], and generates hybrid filtered, static implicit fisheye, and globally scaled views. A node's priority is its *fractal value* which has the following general form:

$$\begin{aligned} FV(focus) &= 1 \\ FV(child(x)) &= (C{N_x}^{-\frac{1}{D}}) \times FV(x) \end{aligned}$$

where N_x is the branching factor of node x, D is a fractal dimension, and $0 < C \leq 1$.

The fractal filtering mechanism has one nice property that the generalized *FEV* of trees [7] does not: it guarantees that as the *FP* changes, the number of nodes whose fractal value exceeds a specified threshold will remain nearly constant regardless of branching factor. Thus if only nodes whose fractal values exceed the threshold are displayed, both the visual complexity of the resulting layouts and the corresponding system response times will remain relatively constant as the *FP* is changed. In addition to this *fractal pruning*, each visible node is resized so that its size is proportional to its fractal value.

PLUM. This system supports the visualization of abstract data in 3-D [21], and generates hybrid implicit fisheye (static and dynamic) and adorned views. Although it can generate a variety of static and dynamic visualizations, PLUM has been aimed primarily at displaying information about software structures and is best at visualizing graphs, as it provides a variety of 3-D graph layout algorithms. PLUM supports visual emphasis in two ways.

First, a client application can control a common set of graphical object properties including stylistic properties, sizing information, and a priority setting. Stylistic properties such as colour, font, fill and line styles can be controlled explicitly, as can the sizes of objects. The priority setting provides a general means for specifying that an object is important and should be emphasized in the display – different graphical objects may respond to the priority setting in different ways.

Second, PLUM's built-in animation support can be used to reflect changes in the display structure. For example, if an object's priority setting is linked to its z position, then increasing its priority could cause the object to be moved from the back of the 3-D display to the foreground. Animation can be used to make this transition smoother enabling the user to perceive the desired change with less cognitive effort.

Layout-independent FEVs. This *EA* generates hybrid filtering fisheye, non-geometric distorting fisheye, and adorned views [19, 20]. While most distorting *FEV EA*s distort normal views, this *EA* uses built-in and client *PA*s to guide the drawing of the *EV*. The *EA* supports nested graphs with multiple variable magnification strength *FP*s, and generates *EV*s by a two step process. First vertex priorities are computed. Next, the graph is drawn by a bottom-up layout algorithm that computes the size and shape of each node by first positioning its contained nodes.[2] An *EV* is generated by uniformly scaling the bounding box of each node in proportion to its priority – thus each node is resized *before* it is positioned by the layout algorithm. The priority of each arc is the average of the priorities of its endpoints, and is used to vary the thickness of link line segments in the drawing.

Although most *EA*s compute *DOI* values by using additive *DOI* functions, this *EA* uses the following multiplicative form:

$$DOI(v) \quad = \quad API(v) \cdot (p \cdot Wt(v) + (1-p) \cdot Prox(v))$$

The functions API, Wt and $Prox$ return non-negative real values: $API(v)$ is v's *API*, $Wt(v)$ is a generic measure of v's proximity to the *FP*s, $Prox(v)$ is an application-specific measure of v's proximity to the *FP*s, and $0 \leq p \leq 1$ reflects the trade-off between generic and domain-specific notions of proximity. The generic measure of proximity is an approximation to a weighted nesting distance to the *FP*s. Both Wt and $Prox$ take into account the magnification strengths of the individual *FP*s.

Continuous Zoom. This technique generates hybrid filtering fisheye and globally scaled views [2], which permit users to view and navigate hierarchically nested graphs by smoothly expanding and shrinking nodes. The technique evolved from the Variable Zoom *EA* [27] and is part of the Intelligent Zoom interface to time-critical systems; unlike the Variable Zoom, however, this technique calculates priorities using four quantities: the node's *API*, alarm state, connectivity to other important nodes, and perceived user interest (determined by node "open" and "close" actions).

4 A Space of Emphasis Techniques

We propose a six-dimensional space of emphasis techniques; these "axes" appear as the six right-most columns in Table 1. We describe the contents of each column next; we compare the techniques, point out common limitations, and discuss future goals in Section 5.

1. *Approach* – common name of the approach and relevant bibliographic citations.

2. *Date* – date of the earliest citation.

3. *Transformation* – types of transformations employed by the *EA*; the types are:

    - *Visualization-to-Visualization* (vv)*:* obtain the *EV* from the normal view.
    - *Graph-to-Visualization* (gv)*):* obtain the *EV* from the graph topology.

4. *Emphasis Technique* – the types of views used to add emphasis to a visualization (see Section 2.3):

    - *Implicit* (i).
    - *Filtered* (f).
    - *Distorted* (d)*:* the normal view geometry can be used (a vv transformation) or may be ignored (a gv transformation).
    - *Adorned* (a).

5. *Priorities* – methods to obtain priorities:

    - *Supplied* (s)*:* (static) priorities are supplied by user or client *PA*.
    - *Built-in API* (A)*: API* is computed by a built-in *PA*.
    - *Built-in Dist* (D)*:* distances (proximities) are computed by a built-in *PA*.
    - *Client API* (a)*:* permit specification of alternate *API PA*s (*i.e., API* is not static).
    - *Client Dist* (d)*:* permit specification of alternate distance (proximity) *PA*s.

6. *Number of Focal Points* – number of *FP*s explicitly supported by the *EA*:

    - *None* (0)*:* no notion of *FP*.
    - *Single* (1)*:* a single *FP*.
    - *Multiple* $(*)$*:* one or more *FP*s.

7. *Animation* – is animation used for emphasis?

8. *Inputs* – class of inputs supported by the *EA*; classes are listed from the least to the most expressive – each class is fully contained in the next more expressive class:

    - *Sequences* (s)*:* one-dimensional data (*e.g.,* time line).
    - *Hierarchies* $(h) \supset s$*:* hierarchical graphs (*e.g.,* a tree or a forest of trees).
    - *Flat graphs* $(g) \supset h$*:* general graphs.
    - *Nested graphs* $(n) \supset g$*:* graphs in which vertices may be (hierarchically) nested.
    - *Beyond nested graphs* $(*) \supset n$*:* any graphical visualization.

[2] This layout strategy permits the generation of composite layouts [9] by assigning a possibly unique layout algorithm to each vertex.

Note that the product of the legal patterns of each column of Table 1 suggests that the space has $4 \cdot 15 \cdot 32 \cdot 3 \cdot 2 \cdot 5 = 57,600$ points. However, some patterns in one column are not independent of the patterns in another (*i.e.,* the axes are not completely orthogonal); determining the exact shape of the reachable space is one topic of future work.

5 Discussion and Future Work

In this section we summarize some of the observations that can be drawn from the contents of Table 1, and cast these into goals for future graph visualization systems.

Goal 1 (Mixed-mode visualization generation) *Emphasis algorithms should be capable of combining existing and transformed layout information with newly-generated layout information.*

The entries in the *Transformation* column are almost mutually exclusive. This means that all techniques, with the exception of three that allow simple forms of client *PA*s, either generate a visualization from domain data, or else transform an existing drawing into a new drawing. Another possibility is to use both approaches simultaneously. For example, a new drawing could contain some parts of the original drawing and others that were generated from scratch.

Goal 2 (Presentation emphasis techniques) *Determine effective presentation strategies to convey emphasis information.*

The distribution of patterns in the *Emphasis Technique* column indicates that to date, most *EA*s can be grouped into roughly three categories: filtering, distorting, and filtering-distorting hybrids.[3] The *Adorned* column is very sparse, indicating that alternative presentation emphasis strategies based on varying other presentation variables (*e.g.,* colour, texture, lighting, motion, audio, etc.) have not been investigated. Filtering and distortion are two very basic techniques – a quick glance at a presentation graphics book (*e.g.,* [31]) is sufficient to conclude that far greater possibilities remain unexplored.

Goal 3 (Client priority algorithms) *Find a flexible high-level language to specify client priority algorithms.*

[3] It is interesting to note that generalized *FEV*s were originally proposed as a filtering mechanism; perhaps the more sophisticated presentation emphasis techniques such as distortion required more advanced display technology, for they were not studied for another five years. Filtering and distortion serve different purposes, however, thus the presence of filtering-distorting hybrids is not surprising.

The a and d subcolumns of the *Priorities* column are virtually empty – just 6 entries out of 44, showing that most techniques do not permit the specification of client *PA*s. Of the ones that do, the *PA*s must typically be implemented in a low-level language – a flexible high-level specification method is required to build visualization tools with flexible generic presentation emphasis facilities. Of the remaining techniques, 5 circumvent this problem somewhat by using client supplied priority values; unfortunately, the "client" is typically the user – expecting the user to manually specify priorities for a large number of elements is unacceptable; furthermore, in this arrangement, changing the set of *FP*s means that the priorities must be recomputed and resubmitted to the visualization tool, possibly incurring additional communication overhead. The other techniques use fixed *PA*s which are domain-specific and non-portable.

Goal 4 (DOI mapping) *Find ways to specify the mapping between priority values and presentation variables.*

Although this property is not reflected in Table 1, it is important to be able to easily describe how differences in priorities should be reflected in the presentation aspects of automatically generated visualizations.

Goal 5 (Animation) *Incorporate "animation" to eliminate or reduce abrupt transitions in presentation variables caused by changes in priority values.*

Only 7 out of 22 techniques provide some type of animation. Animation, when effectively used, can transform a cognitive task into a perceptual one [22]. In fact, studies of interaction with 3-D graph layouts have shown that motion can provide more valuable perceptual clues than stereopsis [1, 33]. Animation is also useful in 2-D; graphical fisheye views [23, 24], for example, use smooth animation to create a dynamic *FEV*: the distortion is continually recomputed and updated as the user changes the location of the *FP* by dragging a mouse.

Goal 6 (Validating effectiveness) *Determine a set of relevant tasks, experimental methods, and validation criteria that could be used to measure the effectiveness of emphasis techniques.*

Another property that is not reflected in Table 1, is an indication of how effective the proposed technique is at visualizing large information spaces. In fact, of all of the techniques reviewed, only two [11, 27] have been formally tested by measuring user performance in solving tasks with the aid of *EV*s. Note that one way to improve the effectiveness of an emphasis technique is to reduce user disorientation by increasing layout stability when the set of *FP*s is changed [3].

Approach	***Date***	***Transfor-mation*** gv/vv	***Emphasis Technique*** $i/f/d/a$	***Priorities*** $s/A/D/a/d$	***# of FPs*** 0/1/$*$	***Ani-ma tion?***	***Inputs*** $s/h/g/n/*$
Non-graph-oriented:							
Generalized FEVs [7]	04/86	■□	□■□□	□□□■■	○◉○	□	▰▰▰▰▰
Aircraft maintenance [18]	08/90	■□	□■□□	□■■□□	○○◉	□	▰▰▰▱▱
Perspective Wall [15]	04/91	■□	□□■□	□□■□□	○◉○	■	▰▱▱▱▱
Tree-Maps [12, 13, 28, 32]	10/91	■□	□□■□	■□□□□	◉○○	□	▰▰▱▱▱
Stretch Tools [25]	09/92	□■	□□■□	□□□□□	○○◉	□	▰▰▰▰▰
Morphing [26]	11/93	□■	□□■□	□□■□□	○○◉	□	▰▰▰▰▰
Graph-oriented:							
Generalized FEVs of trees [7]	04/86	■□	□■□□	□■■□□	○◉○	□	▰▰▱▱▱
SemNet (implicit) [5]	1988	□□	■□□□	□□□□□	○◉○	■	▰▰▰▰▱
SemNet (clustering) [5]	1988	■□	□■□□	□■■□□	○◉○	□	▰▰▰▰▱
Topographic networks [10, 11]	1989	■□	□■■□	□■■□□	○◉○	□	▰▰▰▱▱
Cone trees (implicit/search) [22]	04/91	□□	■□□□	□□□□□	○◉○	■	▰▰▱▱▱
Cone trees (gardening) [22]	04/91	■□	□■□□	□□□□□	○○◉	□	▰▰▱▱▱
MVP display methods [17, 29]	09/91	□■	□□■□	□□■□□	○○◉	□	▰▰▰▰▱
Associative networks [6]	10/91	■□	□■■■	□■■□□	○◉○	□	▰▰▰▱▱
Interactive graph layout [9]	1991	■□	□□■□	■□□□□	◉○○	□	▰▰▱▱▱
Graphical FEVs [23, 24]	05/92	□■	□■■□	□■■■■	○◉○	■	▰▰▰▱▱
Abridgment [16]	05/93	■□	□■■□	■■■□□	○○◉	■	▰▰▰▰▱
Variable Zoom [27]	05/93	□■	□■■□	□□■□□	○○◉	□	▰▰▰▰▱
Fractal views of trees [14]	08/93	■□	■■■□	□□■□□	○◉○	□	▰▰▱▱▱
PLUM [21]	08/93	■□	■□□■	■□□□□	◉○○	■	▰▰▰▰▰
Layout-indep. FEVs [19, 20]	08/93	■■	□□■■	□□■■■	○○◉	□	▰▰▰▰▱
Continuous Zoom [2]	05/94	□■	□■■□	■■■□□	○○◉	■	▰▰▰▰▱
An ideal technique?:							
	?	■■	■■■■	■■■■■	○○◉	■	▰▰▰▰▰

Table 1: A comparison of recent presentation emphasis techniques.

6 Summary

Although presentation emphasis techniques have been studied under a number of guises, the research in this field has been largely fragmented. In this paper we have collected and reviewed a significant portion of recent literature and proposed a common set of criteria which permitted a direct comparison of a number of otherwise disparate approaches. The comparison helped us to identify common shortcomings, which, in turn, suggested several directions for future work.

References

[1] K.W. Arthur, K.S. Booth, and C. Ware. Evaluating 3d task performance for fish tank virtual worlds. *ACM Transactions on Office Information Systems*, 11(3):239–265, Jul. 1993.

[2] L. Bartram, R. Ovans, J. Dill, M. Dyck, A. Ho, and W.S. Havens. Contextual assistance in user interfaces to complex, time-critical systems: The intelligent zoom. In *GI '94: Graphics Interface 1994*, Banff, Alberta, Canada, May. 1994.

[3] P. Eades, W. Lai, K. Misue, and K. Sugiyama. Preserving the mental map of a diagram. Technical Report IIAS-RR-91-16E, Fujitsu Laboratories, Aug. 1991.

[4] P. Eades and R. Tamassia. Algorithms for drawing graphs: an annotated bibliography. Technical Report CS-89-09, Dept. of Comp. Sci., Brown U., 1989.

[5] K.M. Fairchild, S.E. Poltrock, and G.W. Furnas. Semnet: Three-dimensional graphic representations of large knowledge bases. In R. Guindon, editor, *Cognitive Science and its Applications for Human-Computer Interaction*, pages 201–233. Lawrence Erlbaum Associates, 1988.

[6] R.H. Fowler, W.A.L. Fowler, and B.A. Wilson. Integrating query, thesaurus, and documents through

a common visual representation. In *ACM SIGIR '91*, pages 142–151. ACM, Oct. 1991.

[7] G.W. Furnas. Generalized fisheye views. In *ACM CHI '86*, pages 16–23, Boston, MA, Apr. 1986. ACM.

[8] D. Harel. On visual formalisms. *Communications of the ACM*, 31(5):514–530, May. 1988.

[9] T.R. Henry and S.E. Hudson. Interactive graph layout. In *ACM UIST '91*, pages 55–64. ACM, 1991.

[10] J.G. Hollands. Presenting a network graphically: A comparison of performance using fisheye view and scrolling. Master's thesis, Dept. of Psychology, U. of Guelph, Guelph, Ontario, Canada, 1988.

[11] J.G. Hollands, T.T. Carey, M.L. Matthews, and C.A. McCann. Presenting a graphical network: A comparison of performance using fisheye and scrolling views. In G. Salvendy and Smith M.J., editors, *Designing and Using Human-Computer Interfaces and Knowledge-Based Systems*, pages 313–320. Elsevier, 1989.

[12] B. Johnson. Treeviz: Treemap visualization of hierarchically structured information. In *ACM CHI '92*, pages 369–370, Monterey, CA, May. 1992. ACM.

[13] B. Johnson and B. Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *IEEE Visualization '91*, pages 284–291, San Diego, CA, Oct. 1991.

[14] H. Koike and H. Yoshihara. Fractal approaches for visualizing huge hierarchies. In *VL '93: IEEE Symposium on Visual Languages*, pages 55–60, Bergen, Norway, Aug. 1993.

[15] J.D. Mackinlay, G.G. Roberson, and S.K. Card. The perspective wall: Detail and context smoothly integrated. In *ACM CHI '91*, pages 173–179. ACM, Apr. 1991.

[16] K. Misue. D-abductor 2.0 user manual. Technical Report IIAS-RR-93-9E, Fujitsu Laboratories, May. 1993.

[17] K. Misue and K. Sugiyama. Multi-viewpoint perspective display methods: Formulation and application to compound graphs. In *4th Intl. Conf. on Human-Computer Interaction*, volume 1, pages 834–838, Stuttgart, Germany, Sep. 1991. Elsevier.

[18] D.A. Mitta. A fisheye presentation strategy: Aircraft maintenance data. In *INTERACT '90*, pages 875–880. IFIP, Elsevier, Aug. 1990.

[19] E.G. Noik. Exploring large hyperdocuments: Fisheye views of nested networks. In *ACM Hypertext '93*, pages 192–205, Seattle, WA, Nov. 1993.

[20] E.G. Noik. Layout-independent fisheye views of nested graphs. In *VL '93: IEEE Symposium on Visual Languages*, pages 336–341, Bergen, Norway, Aug. 1993.

[21] S.P. Reiss. A framework for abstract 3d visualization. In *VL '93: IEEE Symposium on Visual Languages*, pages 108–115, Bergen, Norway, Aug. 1993.

[22] G.G. Robertson, J.D. Mackinlay, and S.K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In *ACM CHI '91*, pages 189–194. ACM, Apr. 1991.

[23] M. Sarkar and M.H. Brown. Graphical fisheye views of graphs. In *ACM CHI '92*, pages 83–91, Monterey, CA, May. 1992. ACM.

[24] M. Sarkar and M.H. Brown. Graphical fisheye views of graphs. Technical Report 84, DEC System Research Center, Palo Alto, CA, Mar. 1992.

[25] M. Sarkar and S.P. Reiss. Manipulating screen space with *stretch tools:* visualizing large structure on small screen. Technical Report CS-92-42, Dept. of Comp. Sci., Brown U., Providence, RI, Sep. 1992.

[26] M. Sarkar, S.S. Snibbe, and S.P. Reiss. Stretching the rubber sheet: A metaphor for visualizing large structures on small screens. In *ACM UIST '93*. ACM, Nov. 1993.

[27] D. Schaffer, Z. Zuo, L. Bartram, J. Dill, S. Dubs, S. Greenberg, and M. Roseman. Comparing fisheye and full-zoom techniques for navigation of hierarchically clustered networks. In *Graphics Interface '93*, pages 87–96, May. 1993.

[28] B. Shneiderman. Tree visualization with tree-maps: A 2-d space filling approach. *ACM Transactions on Graphics*, 11(1):1–39, Jan. 1992.

[29] K. Sugiyama and K. Misue. 'good' graphic interfaces for 'good' idea organizers. In *Human-Computer Interaction – INTERACT '90*, pages 521–526. IFIP, Elsevier, Aug. 1990.

[30] K. Sugiyama and K. Misue. Visualization of structural information: Automatic drawing of compound digraphs. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(4):876–892, 1991.

[31] E.R. Tufte. *Envisioning Information*. Graphics Press, P.O. Box 430, Cheshire, Connecticut, 06410, 1990.

[32] D. Turo and B. Johnson. Improving the visualization of hierarchies with treemaps: Design issues and experimentation. In *IEEE Visualization '92*, pages 124–131, Boston, MA, 1992.

[33] C. Ware, K. Arthur, and K.S. Booth. Fish tank virtual reality. In *InterCHI'93*, pages 37–42, Amsterdam, Netherlands, May. 1993.

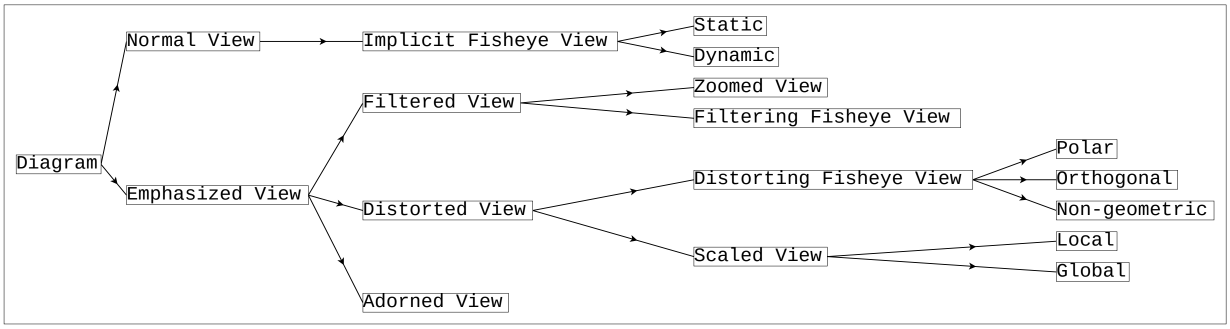
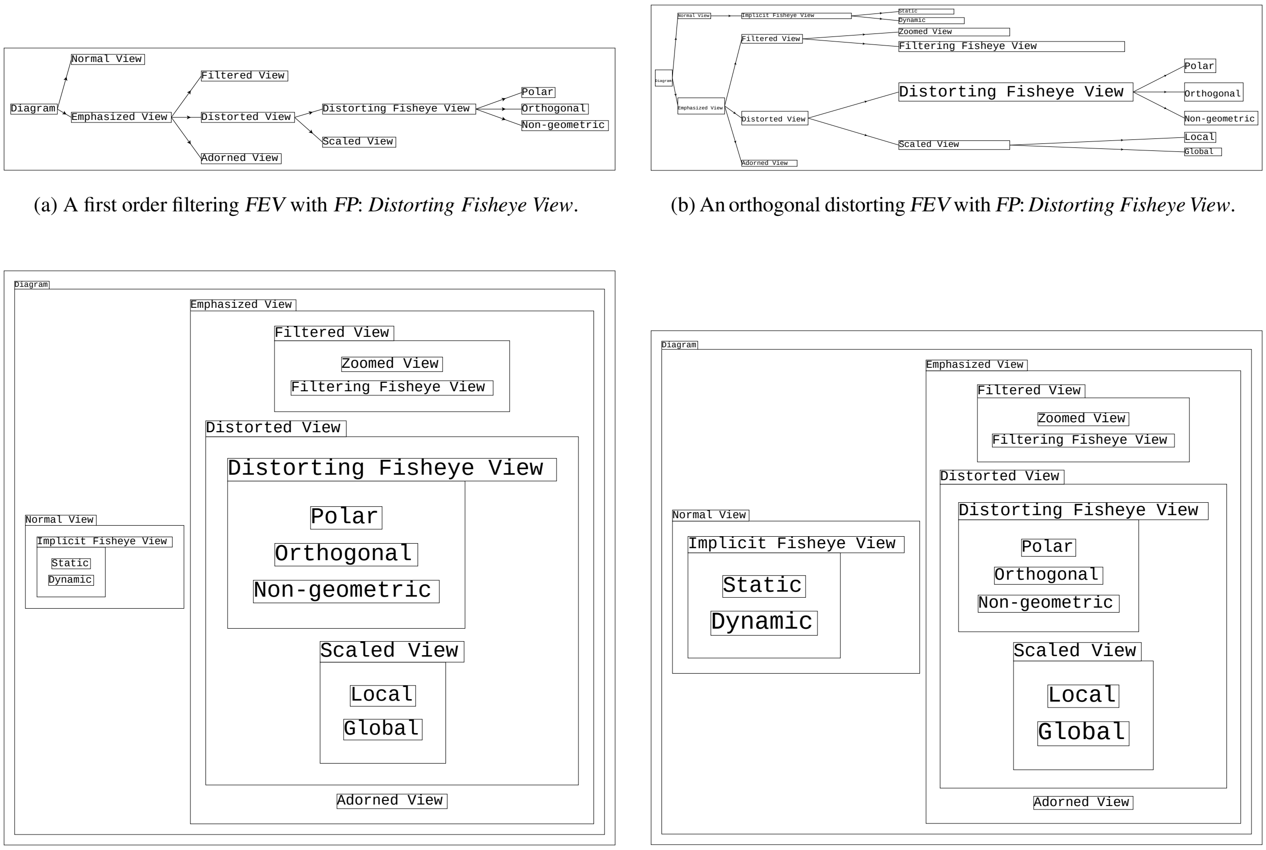


Figure 1: A simple taxonomy of graph views; note that existing categories could be further refined and more categories could be added.



(a) A first order filtering *FEV* with *FP*: *Distorting Fisheye View*.

(b) An orthogonal distorting *FEV* with *FP*: *Distorting Fisheye View*.

(c) A non-geometric (layout-independent) distorting *FEV* with *FP*: *Distorting Fisheye View*.

(d) A non-geometric (layout-independent) distorting *FEV* with *FP*s: *Dynamic* and *Global*.

Figure 2: Several emphasized views of the graph view taxonomy.