Difference Metrics for Interactive Orthogonal Graph Drawing Algorithms

Stina Bridgeman   *Roberto Tamassia*

Center for Geometric Computing
Department of Computer Science
Brown University
Providence, Rhode Island 02912–1910
{ssb,rt}@cs.brown.edu

Abstract

Preserving the "mental map" is a major goal of interactive graph drawing algorithms. Several models have been proposed for formalizing the notion of mental map. Additional work needs to be done to formulate and validate "difference" metrics which can be used in practice. This paper introduces a framework for defining and validating metrics to measure the difference between two drawings of the same graph, and gives a preliminary experimental analysis of several simple metrics.

This version of the paper is suitable for color printing.

1   Introduction

Graph drawing algorithms have traditionally been developed using a batch model, where the graph is redrawn from scratch every time a drawing is desired. These algorithms, however, are not well suited for interactive applications, where the user repeatedly makes modifications to the graph and requests a new drawing. When the graph is redrawn it is important to preserve the look (the user's "mental map" [19]) of the original drawing as much as possible, so the user does not need to spend a lot of time relearning the graph.

The problems of incremental graph drawing, where vertices are added one at a time, and the more general case of interactive graph drawing, where any combination of vertex/edge deletion and insertion is allowed at each step, have been starting to receive more attention. See, for example, the work of Biedl and Kaufmann [3], Brandes and Wagner [4], Bridgeman et. al. [6], Cohen et. al. [8], Fößmeier [12], Miriyala, Hornick, and Tamassia [18], Moen [20], North [21], Papakostas, Six, and Tollis [22], Papakostas and Tollis [23], and Ryall, Marks, and Shieber [26]. However, while the algorithms themselves have been motivated by the need to preserve the user's mental map, much of the evaluation of the algorithms has so far focused on traditional optimization criteria such as the area and the number of bends and crossings (see, for example, the analysis in Biedl and Kaufmann [3], Fößmeier [12], Papakostas, Six, and Tollis [22], and Papakostas and Tollis [23]). Mental map preservation is often achieved by attempting to minimize the change between drawings — typically by allowing only very limited modifications (if any) to the position of vertices and edge bends in the existing drawing — making it important to be able to measure precisely how much the look of the drawing changes. Animation can be used to provide a smooth transition between the drawings and can help compensate for greater changes in the drawing, though it is still important to limit, if not minimize, the difference between the drawings because if there is a very large change it can become difficult to generate a clear, useful animation. It is thus still important to have a measure of how the look of the drawing changes.

Studying "difference" metrics to measure how much a drawing algorithm changes the user's mental map has a number of benefits, including

- providing a basis for studying the behavior of constraint-based interactive drawing algorithms like InteractiveGiotto [6], where meaningful bounds on the movement of any given part of the drawing are difficult to obtain,

- providing a technique to compare the results of different interactive drawing algorithms, and

- providing a goal for the design of new drawing algorithms by identifying which qualities of the drawing are the most important to preserve.

Finding a good difference metric also has an immediate practical benefit, namely solving the "rotation problem" of InteractiveGiotto. Giotto [27], the core of InteractiveGiotto, does not take into account the coordinates of vertices and bends in the
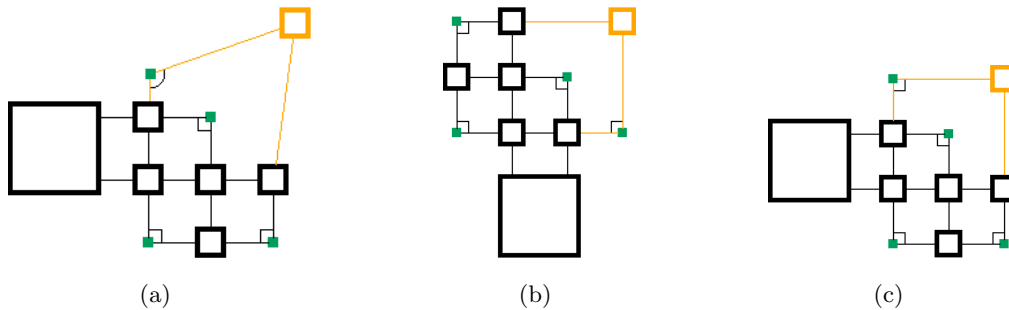
Figure 1: The rotation problem of InteractiveGiotto. (a) is the user-modified graph (the user's changes are shown in orange); (b) and (c) show the uncorrected and corrected outputs, respectively. While (b) and (c) are both clearly drawings of the graph shown in (a), the resemblance is more readily seen in the properly rotated and reflected drawing (c).

original drawing when constructing a new drawing — and, as a result, the output of InteractiveGiotto may be rotated by a multiple of 90 degrees and/or be a mirror-image reflection of the original drawing (Figure 1). The problem can be solved by computing the value of the metric for each of the possible rotations and choosing the rotation with the smallest value.

Eades et. al. [11], Lyons, Meijer, and Rappaport [16], and Misue et. al. [19] have proposed several models for formalizing the notion of the mental map, though more work needs to be done to formally define potential difference metrics and then experimentally validate (or invalidate) them. Validation can be via user studies similar to those done by Purchase, Cohen, and James [24, 25] to evaluate the impact of various graph drawing aesthetics on human understanding.

Motivated by applications to InteractiveGiotto, this paper will focus primarily on difference metrics for orthogonal drawings, though many of the metrics can be used without modification for arbitrary drawings. Section 3 describes several potential metrics, Section 4 presents a framework for evaluating the suitability of the metrics along with a preliminary evaluation, and Section 5 outlines plans for future work.

2   Preliminaries

Paired Sets of Objects   Every metric presented in this paper compares two drawings D and D' of the same graph G. Each object of G has a representation in both D and D'. For example, each vertex of G has a representation in both drawings — if vertices are drawn as rectangles, this representation consists of the position, size, color, line style, etc. of the rectangle. Similarly, each edge of G has a representation in both drawings, and if edges are drawn as polylines, this

representation consists of the positions of the bends and endpoints, plus the color, line style, and so forth.

A *paired set* of objects is a set of pairs describing the representation of each object in the two drawings. The paired set of vertices of G is the set of pairs (r_v, r'_v), where r_v and r'_v are the representations of v in D and D', respectively, for all vertices v of G. The paired set of edges is defined similarly. Referring to a paired set is simply a way of matching up the elements of each drawing according to the underlying object of G that they represent.

It should be noted that the only features of the representations that are considered here are the geometric features such as position and size; other features like vertex color and line style are also very important in preserving the look of the drawing and may be able to at least partially compensate for geometric changes but are not considered further at this stage.

Point Set Selection   Most of the metrics are based on point sets, working with paired sets of points derived from the edges and vertices of the graph rather than the edges and vertices themselves. Once derived, each point is independent from the others — there is no notion of a group of points being related because they were derived from the same vertex, for example.

Points can be selected in a number of ways. North [21] suggests that vertex positions are a significant visual feature of the drawing, and two vertex-centered methods — *centers* and *corners* — are used here to reflect that idea. "Centers" consists of the center points of each vertex; this captures how vertices move. "Corners" uses the four corners of each vertex, taking into account both vertex motion (the movement of the center) and changes in the vertex dimension. It seems important to take into account changes in vertex dimension because a vertex with a large or distinctive shape can act as a landmark to orient the user to the drawing; loss of that landmark makes orientation more difficult. Other choices of points can include edge bends and endpoints.

Points can also be derived from groups of graph objects. For example, the vertices of the graph can be partitioned, and points derived from the centroids or convex hull of the partitions. Point sets based on partitioning can be used to capture information about larger units of the graph, such as groups of vertices representing related objects.

Drawing Alignment   The key features of a graph object's representation are coordinates, which means metrics may be sensitive to the particular values of those coordinates — the scaling and translation of one point set relative to the other can make a large difference in the value of the metric (Figure 2).

To eliminate this effect, the drawings are *aligned* before coordinate-sensitive metrics are computed. This is done by extracting a (paired) set of points from the drawings and applying a point set matching algorithm to obtain the best fit. In
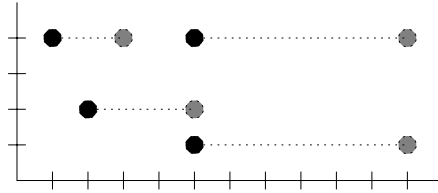
Figure 2: Two point sets (black and gray) superimposed. (Corresponding points in the two sets are connected with dotted lines.) As shown, the Euclidean distance metric (Section 3.1) would report a distance of 4.25. However, translating the gray points one unit to the left and then scaling by $1/2$ in the x direction allows the point sets to be matched exactly, for a distance of 0. It should be noted that exact matches are not possible in general.

general the matching algorithm should take into account scaling, translation, and rotation, though it may be possible to eliminate one or more of the transformations for certain metrics or if something is known about the relationship between the two drawings. For example, interactive drawing algorithms often preserve the rotation of the drawing (see Biedl and Kaufmann [3] and Papakostas and Tollis [23] for examples), eliminating the need to consider rotation in the alignment stage. Even if the algorithm does not preserve the rotation, for orthogonal drawings there are only eight possible rotations for the second drawing relative to the first — four multiples of $\pi/2$, applied to the original drawing and its reflection about the x axis — which can be handled by computing the metric separately for each rotation and taking the minimum value instead of incorporating rotation into the alignment process.

A great deal of work has been done on point set matchings; see Alt, Aichholzer, and Rote [1], Chew et. al. [7], and Goodrich, Mitchell, and Orletsky [14] for several methods of obtaining both optimal and approximate matchings. Different methods can be applied when the correspondence between points is known as it is here; Imai, Sumino, and Imai [15] provide an algorithm that minimizes the maximum distance between corresponding points under translation, rotation, and scaling. In the implementation used in Section 4, the alignment is performed by using gradient search to minimize the distance squared between points.

3 Metrics

The difference metrics being considered fall into six categories:

- **distance**: metrics based on the distance between points or the distance points move between drawings
- **proximity**: metrics based on the nearness of points and the clustering of points according to the distance between them

- **partitioning**: metrics based on partitioning points according to measures other than proximity
- **orthogonal ordering**: metrics based on the relative angle between pairs of points
- **shape**: metrics based on the sequence of horizontal and vertical segments of the graph's edges
- **topology**: metrics based on the embedding of the graph

Proximity, ordering, and topology are suggested by Eades et. al. [11], Lyons, Meijer, and Rappaport [16], and Misue et. al. [19] as qualities which should be preserved; distance (also suggested by Lyons, Meijer, and Rappaport [16]), shape, and partitioning reflect intuition about what causes drawings to look different. Within each category specific metrics were chosen to capture intuition about what qualities of the drawing are important to preserve.

An alternative taxonomy is given by Biedl et. al. [2]. This taxonomy is similar to the one given above, with the main distinctions being the inclusion of "feature similarity" based on the appearance of regions of the drawing, and the grouping of all measures based on the comparison of point sets into a single "metric similarity" category.

In the following, let P denote the (paired) set of points, and p_i and p_i' be the coordinates of point i in drawings D and D', respectively. Also let $d(p, q)$ be the Euclidean distance between points p and q.

3.1   Distance

The distance metrics reflect the simple observation that drawings that look very different cannot be aligned very well, and vice versa. Since the alignment is based on distance minimization, these metrics essentially measure the quality of the alignment.

In order to make the value of the distance metrics comparable between pairs of drawings, they are scaled by the graph's *unit length u*. For orthogonal drawings the unit length can be computed by taking the greatest common divisor of the Manhattan distances between vertex centers and bend points on edges. Non-orthogonal portions of the drawing, such as modifications of an orthogonal drawing made by the user, can be ignored during the computation. While the determination of the unit length will be unreliable if only a small portion of the drawing is orthogonal, scaling by the unit length is not necessary in some applications (e.g., solving the rotation problem of InteractiveGiotto) and can often be supplied manually if it is required (e.g., the drawing algorithm is known to place vertices on a unit grid).

Hausdorff Distance   The *undirected Hausdorff distance* is a standard metric for determining the distance between two point sets, and measures the largest distance

between a point in one set and its nearest neighbor in the other.

$$\text{haus}(D, D') = \frac{1}{u} \max\{\max_i \min_j d(p_i, p'_j), \max_i \min_j d(p'_i, p_j)\}$$

where $1 \leq i, j \leq |P|$ and $j \neq i$.

Euclidean Distance *Euclidean distance*, used by Lyons, Meijer, and Rappaport [16], is a simple metric measuring the average distance moved by each point from the first drawing to the second; it is motivated by the notion that if points move a long way from their locations in the first drawing, the second drawing will look very different.

$$\text{dist}(D, D') = \frac{1}{u\,|P|} \sum_{1 \leq i \leq |P|} d(p_i, p'_i)$$

Relative Distance *Relative distance* measures the average change in the distance between each pair of points between the first drawing and the second. This measures how much the points in each drawing move relative to each other; it is similar in some respects to the orthogonal ordering metrics in Section 3.4.

$$\text{rdist}(D, D') = \frac{1}{|P|\,(|P| - 1)} \sum_{1 \leq i, j \leq |P|} \left| d(p_i, p_j) - d(p'_i, p'_j) \right|$$

3.2 Proximity

The proximity metrics reflect the idea that points near each other in the first drawing should remain near each other in the second drawing. This is stronger than the distance metrics because it captures the idea that if an entire subgraph moves relative to another (and there are only small changes within each subgraph), the distance should be less than if each point in one of the subgraphs moves in a different direction (Figure 3).

Three different metrics are used to try to capture this idea: nearest neighbor within (nn-within), nearest neighbor between (nn-between), and ϵ-clustering.

Nearest Neighbor Within *Nearest neighbor within* is based on the reasoning that if p_j is the closest point to p_i in D, then p'_j should be closest point to p'_i in D'. Considering only distances within a single drawing means that nn-within is alignment-independent and thus not subject alignment errors, but means that it is not suitable for solving the rotation problem of InteractiveGiotto.

This metric has two versions, *weighted* and *unweighted*. In the weighted version the number of points closer to p'_i than p'_j is considered, whereas in the unweighted version only whether or not p'_j is the closest point matters. The reasoning behind the weighted version is that if there are more points between p'_i and p'_j, the visual
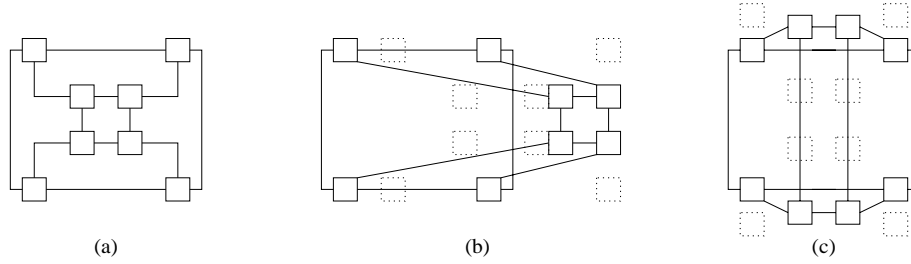
Figure 3: Proximity: (b) looks more like (a) than (c) does because the relative shape of both the inner and outer squares are preserved even though the distance (using the Euclidean distance metric) between (c) and (a) is smaller. An aligned version of the vertices of (a), used in the computation of the distance metric, is shown with dotted lines in (b) and (c).

linkage between p'_i and p'_j has been disrupted to a greater degree and the drawing looks more different.

In both cases the distance is scaled by the number of points being considered and W, the maximum weight contributed by a single point, so that the metric's value is always in the range $[0, 1]$.

$$\text{nn-w}(D, D') = \frac{1}{W\,|P|} \sum_{1 \leq i \leq |P|} \text{closer}(p'_i, p'_j)$$

where p_j is the closest point to p_i in D and

$$\text{closer}(p'_i, p'_j) = \left| \{k \mid d(p'_i, p'_k) < d(p'_i, p'_j)\} \right|, \qquad W = |P| - 2 \qquad \text{(weighted)}$$

$$\text{closer}(p'_i, p'_j) = \begin{cases} 0 & \text{if } d(p'_i, p'_j) \leq d(p'_i, p'_k), k \neq i \\ 1 & \text{otherwise} \end{cases}, \quad W = 1 \qquad \text{(unweighted)}$$

Nearest Neighbor Between   *Nearest neighbor between* is similar to nn-within but instead measures whether or not p'_i is the closest of the points in D' to p_i when the two drawings are aligned. The idea that a point should remain nearer to its original position than any other is also the force behind layout adjustment algorithms based on the Voronoi diagram [16].

$$\text{nn-b}(D, D') = \frac{1}{W\,|P|} \sum_{1 \leq i \leq |P|} \text{closer}(p_i, p'_i)$$

where

$$\text{closer}(p_i, p'_i) = \left| \{j \mid d(p_i, p'_j) < d(p_i, p'_i)\} \right|, \qquad W = |P| - 1 \qquad \text{(weighted)}$$

$$\text{closer}(p_i, p'_i) = \begin{cases} 0 & \text{if } d(p_i, p'_i) \leq d(p_i, p'_j), j \neq i \\ 1 & \text{otherwise} \end{cases}, \quad W = 1 \qquad \text{(unweighted)}$$

Unlike nn-within, nn-between is not alignment- and rotation-independent and thus is suitable for solving the rotation problem.

ϵ-Clustering   The definition for an ϵ-cluster is from Eades et. al [11]: An ϵ-cluster for a point p_i is the set of points p_j such that $d(p_i, p_j) \leq \epsilon$, where a reasonable value to use for ϵ is

$$\epsilon = \max_i \min_{j \neq i} d(p_i, p_j)$$

The ϵ-*cluster* metric measures how the ϵ-cluster for p_i compares to that for p'_i. Let $C_D = \{(i,j) \mid d(p_i, p_j) \leq \epsilon_D\}$ and $C_{D'} = \{(i,j) \mid d(p'_i, p'_j) \leq \epsilon_{D'}\}$. Then

$$\text{clust}(D, D') = 1 - \frac{|C_D \cap C_{D'}|}{|C_D \cup C_{D'}|}$$

The idea is that points should be in the same ϵ-cluster in both drawings.

3.3   Partitioning

The partitioning metrics are based on dividing the point set into subsets according to some criteria, and measuring qualities of these partitions. The motivation for this is to capture "visual units" that the user may use for orientation when learning the new drawing.

Fixed Relative Position Partitioning   A variety of partitioning methods are possible. A simple one is to divide the point set so that the points in each group have the same relative position in both drawings. This identifies blocks of the graph that are the same in both drawings — the larger the partitions, the more unchanged parts and the more similar the drawings.

Two metrics are computed, the average partition size and the number of partitions. Both are scaled to have values between 0 and 1:

$$\text{alsz}(D, D') = 1 - \frac{\left(\dfrac{1}{k} \displaystyle\sum_{1 \leq i \leq k} |S_i|\right) - 1}{|P| - 1} \qquad \text{[average size]}$$

$$\text{alct}(D, D') = \frac{k - 1}{|P| - 1} \qquad \text{[number of partitions]}$$

where the set of partitions is $\{S_1, \ldots, S_k\}$. The idea is that as the drawings become more different, the partition size will decrease and the number of partitions will increase.

The partitioning method and the metrics computed are obviously quite simple, and can be made much more sophisticated. For example, the partitions can be

adjusted to only include points that are also close physically; while it tends not to be the case that points from widely separated regions of the drawing are in the same partition, it can occur. A large distance between points interferes with their grouping as a single visual unit.

Additional sophistications can address things such as how visually separate two adjacent partitions are, since if they are very near or entertwined in one drawing, it is more difficult for the observer to distinguish them and use them as landmarks for the other drawing. Partitions can also be weighted to take into account how well the partition reflects a distinct unit of the graph, so that partitions containing only points derived from a connected subgraph are better than those containing points from several unconnected portions of the graph, even if those subgraphs are physically close together.

3.4 Orthogonal Ordering

The *orthogonal ordering* metric reflects the desire to preserve the relative ordering of every pair of points — if p_i is northeast of p_j in D, p_i' should remain to the northeast of p_j' in D' (Eades et. al. [11] and Misue et. al. [19]). The simplest measurement of difference in the orthogonal ordering is to take the angle between the vectors $p_j - p_i$ and $p_j' - p_i'$ (*constant-weighted* orthogonal ordering). This has the desirable feature that if p_j is far from p_i, $d(p_j, p_j')$ must be larger to result in the same angular move, which reflects the intuition that the relative position of points near each other is more important that the relative position of points that are far apart.

However, simply using the angular change fails to take into account situations such as that shown in Figure 4. This problem can be addressed by introducing a weight that depends on the particular angles involved in the move in addition to size of the move (*linear-weighted* orthogonal ordering).

$$\text{order}(D, D') = \frac{1}{W\,|P|} \sum_{1 \le i,j \le |P|} n \min(\text{order}(\theta_{ij}, \theta_{ij}'), \text{order}(\theta_{ij}', \theta_{ij}))$$

where θ_{ij} is the angle from the positive x axis to the vector $p_j - p_i$, θ_{ij}' is the angle from the positive x axis to the vector $p_j' - p_i'$, and

$$\text{order}(\theta_{ij}, \theta_{ij}') = \int_{\theta_{ij}}^{\theta_{ij}'} \text{weight}(\theta)\, d\theta, \quad W = \min\left\{\int_0^\pi \text{weight}(\theta)\, d\theta, \int_\pi^{2\pi} \text{weight}(\theta)\, d\theta\right\}$$

The weight functions are

$$\text{weight}(\theta) = \begin{cases} \frac{\frac{\pi}{2} - (\theta \bmod \frac{\pi}{2})}{\frac{\pi}{4}} & \text{if } (\theta \bmod \frac{\pi}{2}) > \frac{\pi}{4} \\ \frac{\theta \bmod \frac{\pi}{2}}{\frac{\pi}{4}} & \text{if } (\theta \bmod \frac{\pi}{2}) \le \frac{\pi}{4} \end{cases} \qquad \text{(linear-weighted)}$$

$$\text{weight}(\theta) = 1 \qquad\qquad\qquad\qquad\qquad\qquad \text{(constant-weighted)}$$

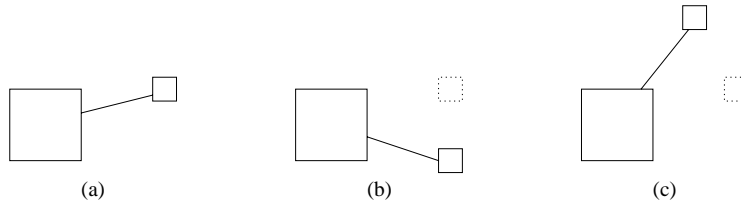(a)   (b)   (c)

Figure 4: Orthogonal ordering: Even though the angle the vertex moves relative to the center of the large vertex is the same from (a) to (b) and from (a) to (c), the perceptual difference between (a) and (c) is much greater. The original location of the vertex is shown with a dotted box in (b) and (c) for comparison purposes.

The λ-matrix model for measuring the difference of two point sets, used by Lyons, Meijer, and Rappaport [16], is based on the concept of order type of a point set, from Goodman and Pollack [13]. This model tries to capture the notion of the relative position of vertices in a straight-line drawing and is thus related to the orthogonal ordering metric.

3.5   Shape

The *shape* metric is motivated by the reasoning that edge routing may have an effect on the overall look of the graph (Figure 5). The shape of an edge is the sequence of directions (north, south, east, and west) traveled when traversing the edge; writing the shape as a string of N, S, E, and W characters yields the *shape string* of the edge. For non-orthogonal edges the direction is taken to be the most prominent direction; for example, if the edge goes from (1,1) to (4,2) the most prominent direction is east. For each pair of edges (e_i, e_i'), the edit distance between the corresponding shape strings is computed. Two methods are used for determining the edit distance. One uses dynamic programming to compute the minimum number of insertions, deletions, or replacements of characters needed to transform one string into the other. The other is similar, but normalizes the measure according to the length of the strings; the algorithm is given by Marzal and Vidal [17]. The value of the shape metric is the average edit distance over the graph's edges.

$$\text{shape}(D, D') = \frac{1}{|E|} \sum_{1 \leq i \leq |E|} \text{edits}(e_i, e_i')$$

Shape is scale- and translation-independent.

The shape metric is similar in spirit to the cost function used by Brandes and Wagner [5] in their dynamic extension of Giotto [27]. Their cost function counts the number of changes in angles at vertices and edge bends; the shape metric takes this in account to some degree by noting changes in the direction of an edge segment.
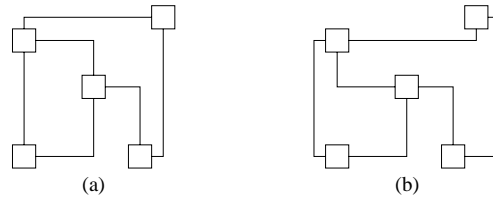
Figure 5: Shape: (a) and (b) look different even though the graphs are the same and the vertices have the same coordinates.

3.6   Topology

The topology metric reflects the idea that preserving the order of edges around a vertex is important in preserving the mental map (Eades et. al. [11] and Misue et. al. [19]) — comparing the drawing produced by Giotto in Figures 6 and 7 to the user's input illustrates this. However, since most interactive orthogonal drawing algorithms always preserve topology, it is not useful as a means of comparing these algorithms. (See, for example, the algorithms of Bridgeman et. al. [6], Biedl and Kaufmann [3], Fößmeier [12], Papakostas, Six, and Tollis [22], and Papakostas and Tollis [23].) Topology is also alignment-independent and so can not be used to solve the rotation problem of InteractiveGiotto. As a result, it is not discussed in any more detail here.

4   Analyzing the Metrics

Once defined, the suitability of the metrics must be evaluated. A good metric for measuring the difference between drawings should satisfy the following three requirements:

- it should *qualitatively* reflect the visual difference between two drawings, i.e. the value increases as the drawings diverge;

- it should *quantitatively* reflect the visual difference so that the magnitude of the difference in the metric is proportional to the perceived difference; and

- in the rotation problem of InteractiveGiotto, the metric should have the smallest value for the correct rotation, though this requirement can be relaxed when the difference between drawings is high since in that case there is no clear "correct" rotation.

The third point is the easiest to satisfy — in fact, most of the metrics defined in the previous section can be used to solve the rotation problem — but is still important worth considering since the problem was one of the factors that first inspired this work.

The running time of the metrics is not considered at this point. While efficiency is clearly a concern, the goal at this stage is to identify the type of measure that

best captures visual similarity. Once this has been done, efficient implementations and/or approximations can be considered.

Some preliminary work has been done on evaluating the proposed metrics with respect to the first and third criteria. Evaluating the qualitative behavior of potential metrics requires a human-generated master ordering of pairs of drawings based on the visual difference between the existing drawing and the new drawing in each pair. This is very difficult to do when each pair of drawings is of a different graph, and most interactive drawing algorithms only produce a single drawing of a particular user-modified graph. InteractiveGiotto, however, makes it possible to obtain a series of drawings of the same input by relaxing the constraints preserving the layout. By default InteractiveGiotto preserves edge crossings, the direction (left or right) and number of bends on an edge, and the angles between consecutive edges leaving a vertex. Recent modifications allow the user to turn off the last two constraints on an edge-by-edge or vertex-by-vertex basis, making it possible to produce a series of drawings of the same graph by relaxing different sets of constraints. A smooth way of relaxing the constraints is to use a breadth-first ordering, expanding outward from the user's modifications. In the first step all of the constraints are applied, in the second step the bend and angle constraints are relaxed for all of the modified objects, in the third step the angle constraints are relaxed for all vertices adjacent to edges whose bends constraints have been relaxed, in the fourth step the bend constraints are relaxed for all edges adjacent to vertices whose angle constraints have been relaxed, and so on, alternating between angle and bend constraints until all of the constraints have been relaxed. This relaxation method is based on the idea that the user is most willing to allow restructuring of the graph near where her changes were made and so the drawings produced resemble drawings that an actual user might encounter. The result is a series of drawings of the same graph — a *relaxation sequence* — bearing varying degrees of similarity to the original.

4.1   An Example

Figures 6 and 7 show portions of two relaxation sequences produced by InteractiveGiotto; the base graphs and user modifications are those used in the first two steps of Figure 2 in Bridgeman et. al. [6]. Giotto's redraw-from-scratch drawing of the graph is also included for comparison. Figures 8 and 9 show the results of several metric-and-pointset combinations for each sequence of drawings. Since Giotto and InteractiveGiotto do not preserve the orientation of the original drawing, each metric is computed for the eight possible rotations (four multiples of $\pi/2$, with or without a reflection around the x-axis) and the lowest value chosen. The color of each column indicates the *confidence*, a measure of how much lower the metric's value is for the best rotation as compared to the second best; red is the most confident and purple is the least confident, with white indicating that the metric is rotation-independent (and so confidence is meaningless) and black indicating that two rotations had the same lowest value. The shading of the column indicates
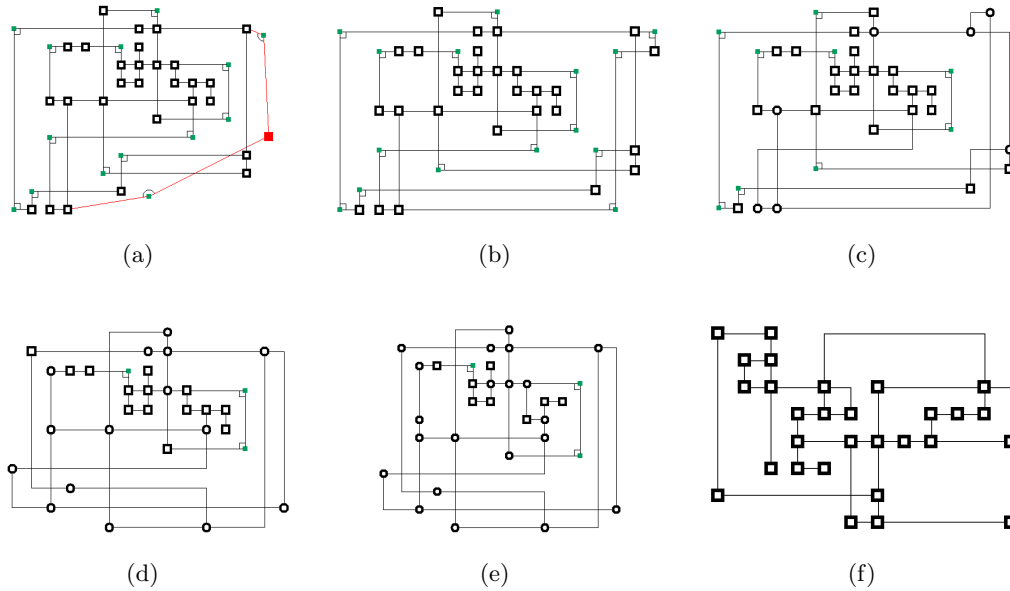
Figure 6: Relaxations for stage 1. (a) shows the user's modifications; (b)–(e) show the output from InteractiveGiotto for relaxation steps 0, 4, 6, and 7; (f) is the output from Giotto. (Intermediate relaxation steps produced drawings identical to those shown and have been left out.) Rounded vertices and bends without markers indicate vertices and bends for which the constraints have been relaxed.

whether or not the metric chose the right rotation — hashing means that the wrong rotation had the lowest value. The correct rotation is defined to be the rotation chosen by the metric/pointset combination with the highest confidence; in practive this is nearly always the rotation a human would pick as the correct answer.

The point sets shown in Figures 8 and 9 show use the following abbreviations:

- **centers**: vertex centers
- **corners**: vertex corners
- **hulls (cen)**: the vertex centers point set is partitioned using fixed relative partitioning; the points used are the points on the convex hull of each partition
- **hulls (cor)**: the vertex corners point set is partitioned using fixed relative partitioning; the points used are the points on the convex hull of each partition
- **centroids (cen)**: the vertex centers point set is partitioned using fixed relative partitioning; the points used are the centroids of each partition
- **centroids (cor)**: the vertex corners point set is partitioned using fixed relative partitioning; the points used are the centroids of each partition

The t or f in brackets following the point set name indicates whether or not points
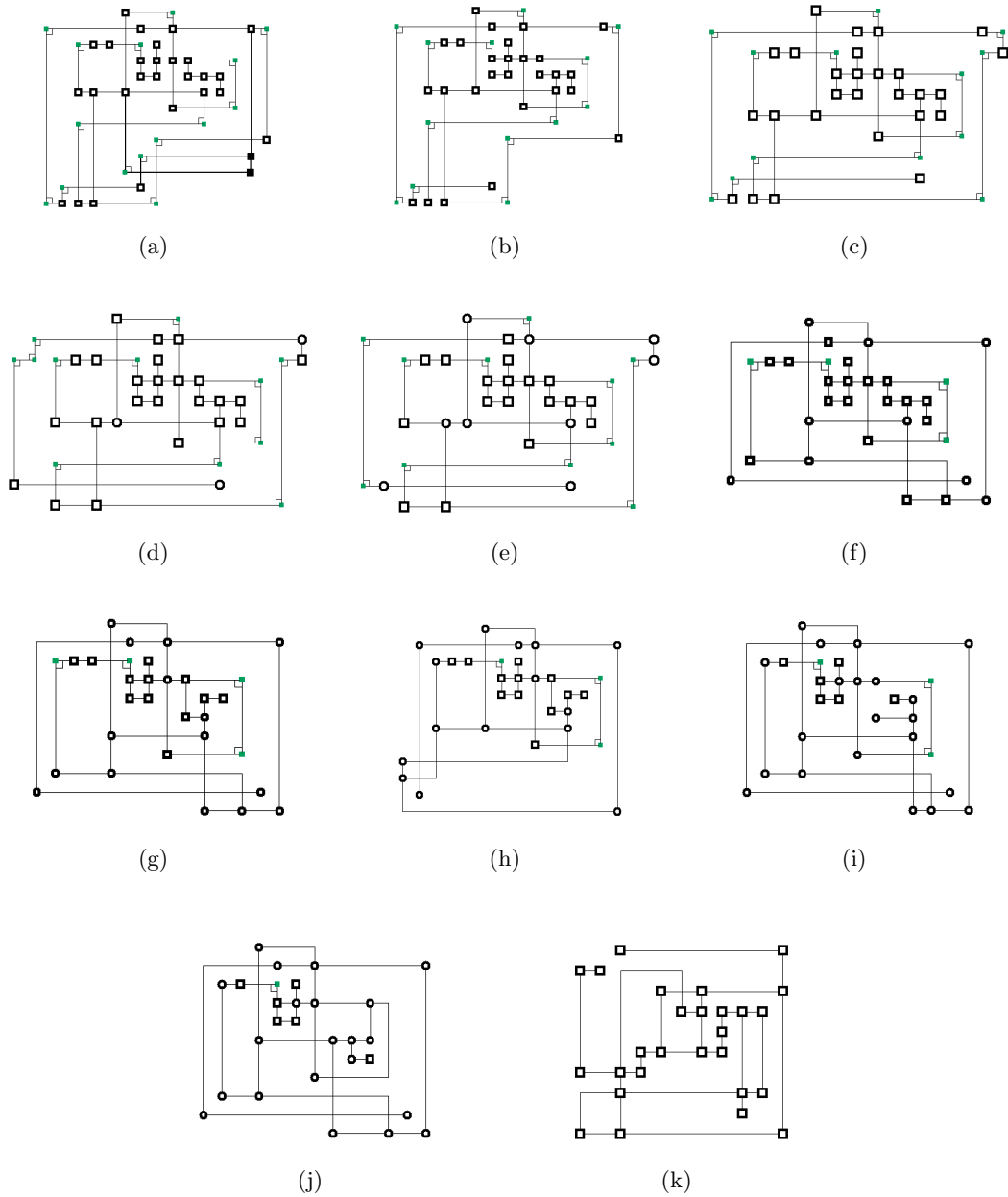
Figure 7: Relaxations for stage 2. (a) shows the starting graph, (b) shows the user's modifications (two vertices and their adjacent edges deleted), (c) is the fully-constrained output from InteractiveGiotto (step 0), (d)–(j) show the output from InteractiveGiotto for relaxation steps 2–8 (step 1 produced the same drawing as step 0), and (k) is the output from Giotto. Rounded vertices and bends without markers indicate vertices and bends for which the constraints have been relaxed.

derived from parts of the graph modified by the user are included in the point set. The first position indicates the value for the point set used for the computation of the metric, the second the point set used for drawing alignment.

The relaxed drawings are labelled as follows:

- n**i**: InteractiveGiotto's drawing of the nth relaxation step
- **xg**: Giotto's output

4.2   Experimental Setup

For the experimental study, a set of 14 graphs with 30 vertices each were extracted from the 11,582-graph test suite used in the experimental studies of Di Battista et. al. [9, 10]. 30-vertex graphs were chosen as being small enough to work with easily, but large enough so that one modification does not affect the entire graph. Future experiments will consider graphs of different sizes.

A random modification was applied to each graph to simulate a user's modification. A modification is one of:

- *edge insert*: insertion of a single edge between two randomly chosen vertices
- *edge delete*: deletion of a single edge
- *edge split*: insertion of a single vertex at the midpoint of an existing edge, splitting the edge into two new edges
- *vertex insert*: insertion of a vertex along with a number of adjacent edges; both the number of edges and their endpoints is chosen randomly
- *vertex delete*: deletion of a vertex and its incident edges

Operations were not allowed to disconnect the graph. Since InteractiveGiotto preserves the embedding of the graph and the edge crossings and bends, new edges were routed so as to mimic how an actual user might draw the edge, instead of simply connecting the endpoints with a straight line (and potentially introducing many edge crossings). Only a single modification was made in each graph because as the user's changes affect a larger portion of the graph, the difference between the new drawings and the original rapidly becomes high and it is difficult to determine an ordering of the relaxation sequence.

Modifications of each type were applied to each of the 14 original graphs, though due to some difficulties with Giotto, the total number of modified graphs generated was only 63.

For each modified graph, a series of progressively relaxed drawings was obtained by incrementally relaxing the constraints given to InteractiveGiotto. The number of relaxed drawings for each modified graph ranged from 8 to 17, but was generally around 11. Each sequence of drawings was then ordered by a human according to increasing visual distance from the original drawing and this ordering was compared to the orderings produced by sorting the drawings in each relaxation sequence according to the computed values of the metric. The orderings were compared using
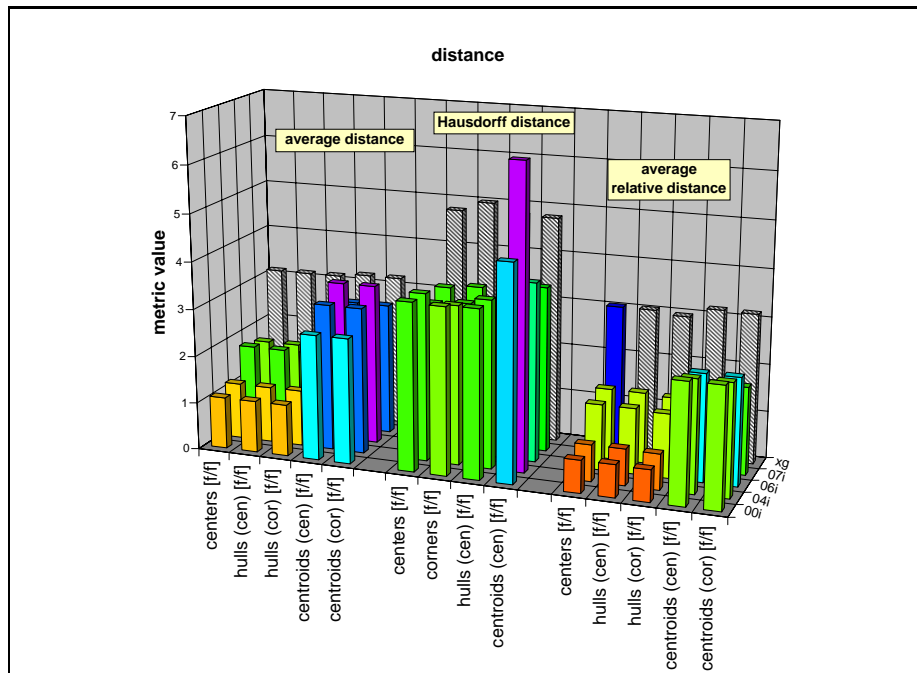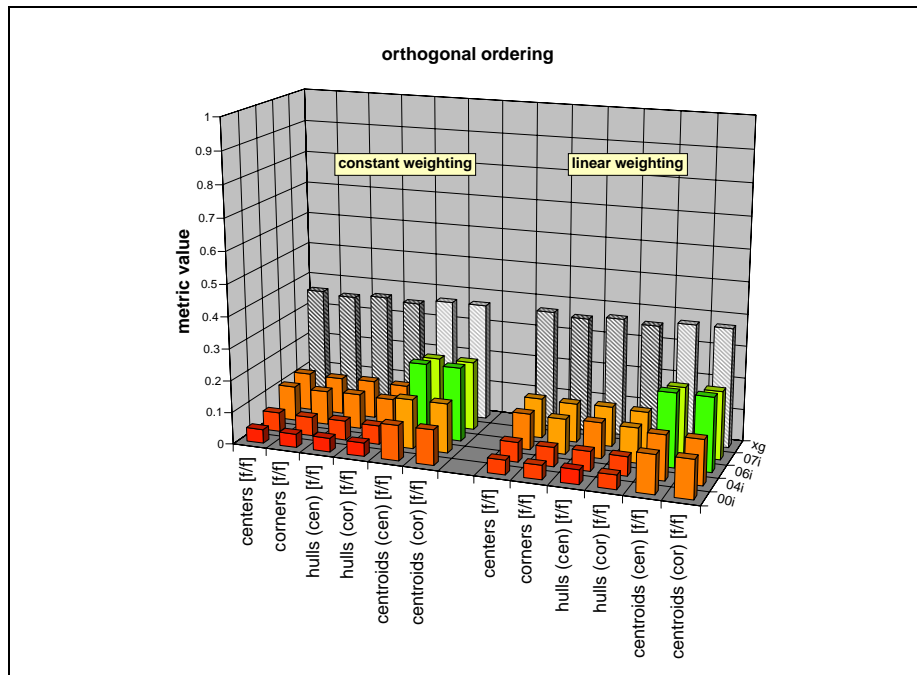
Figure 8: Selected metric values for each drawing in stage 1.
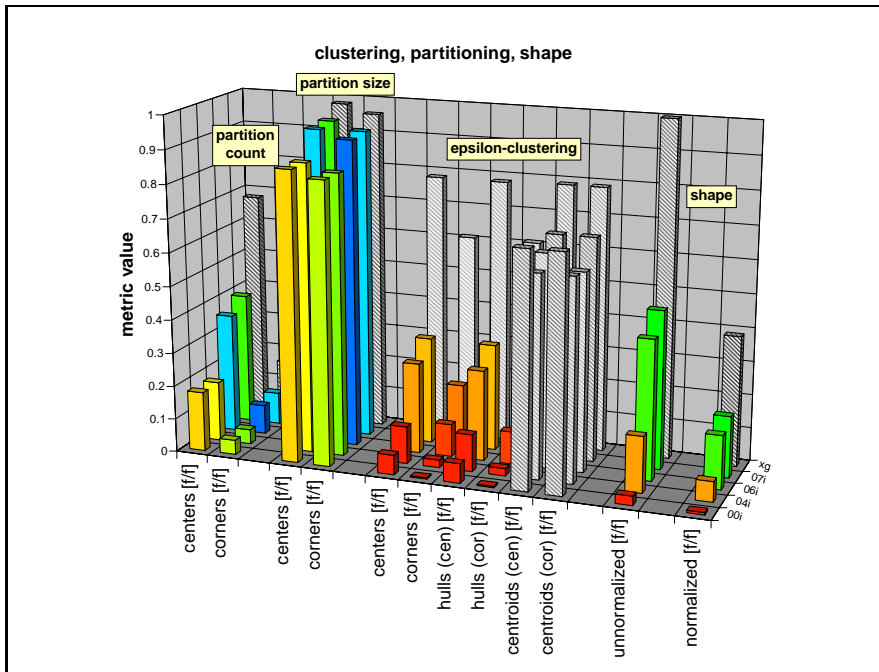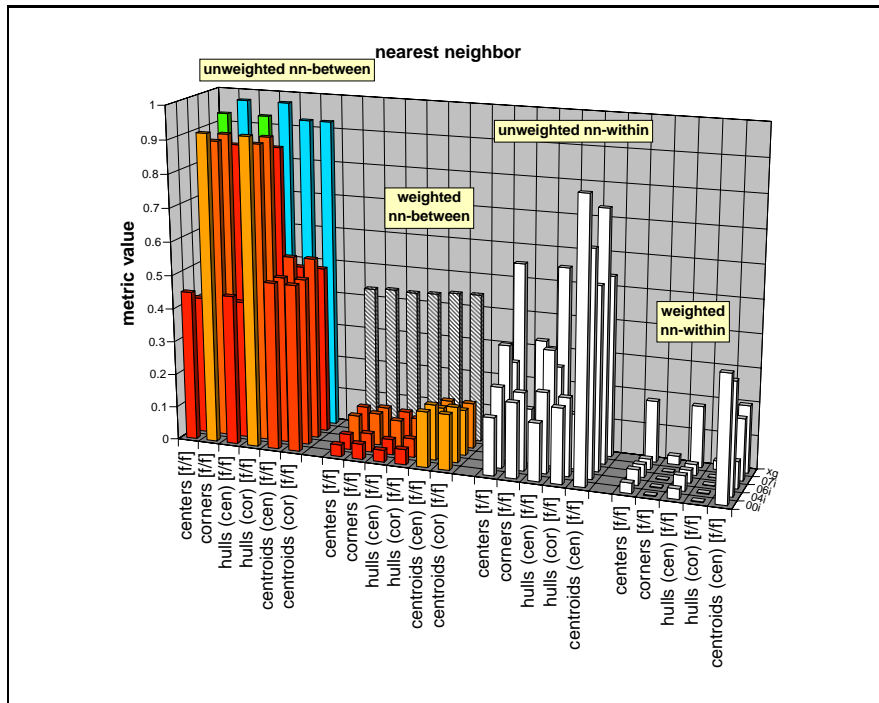
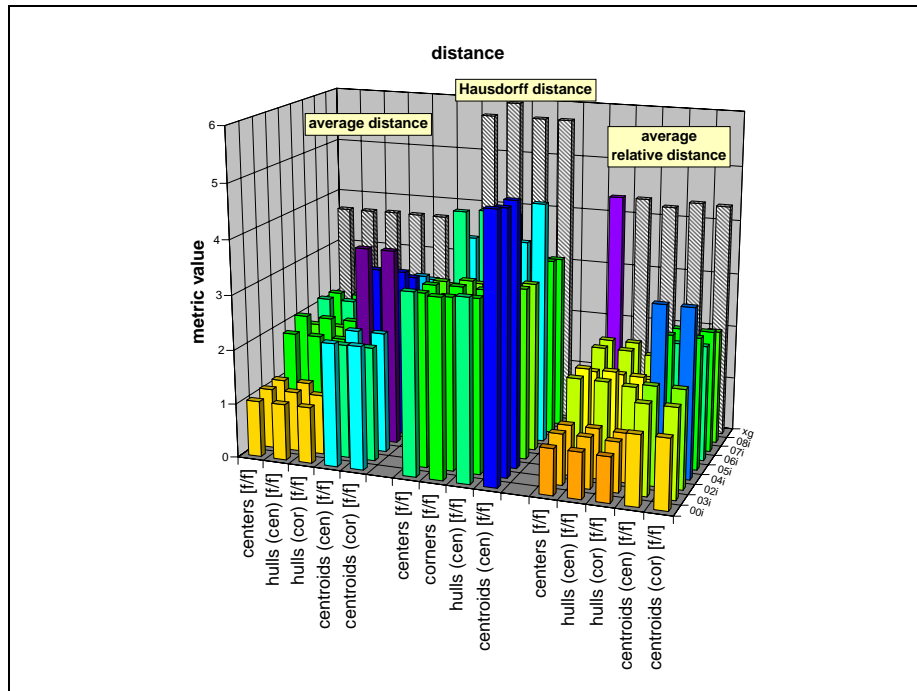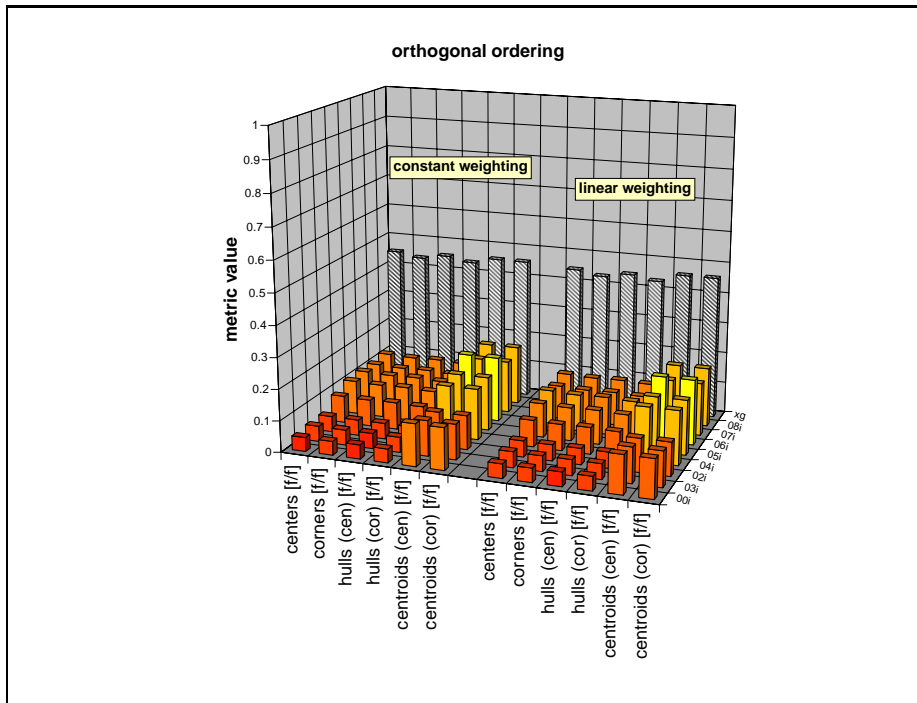Figure 8: Selected metric values for each drawing in stage 1. (continued)

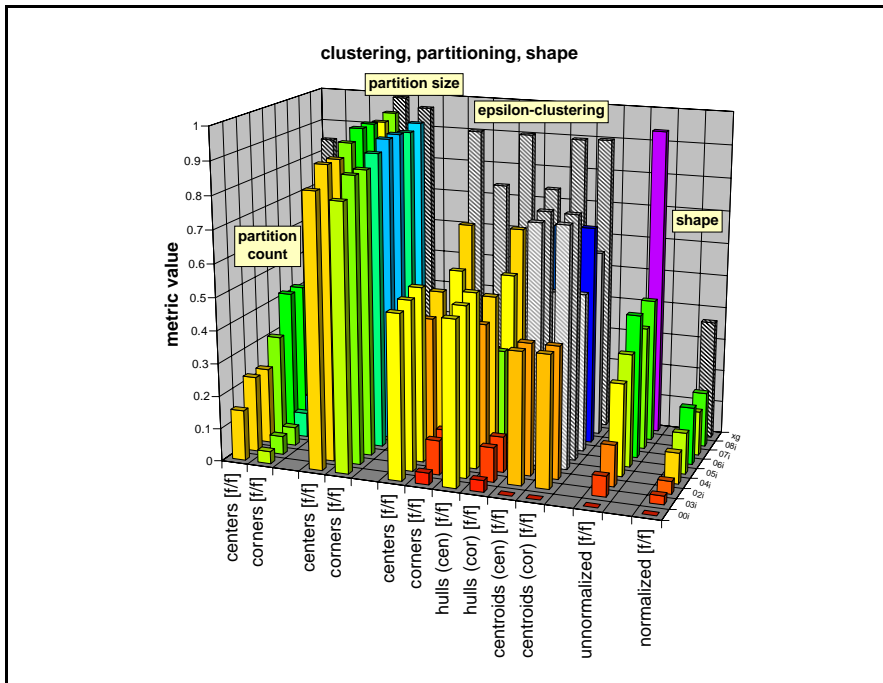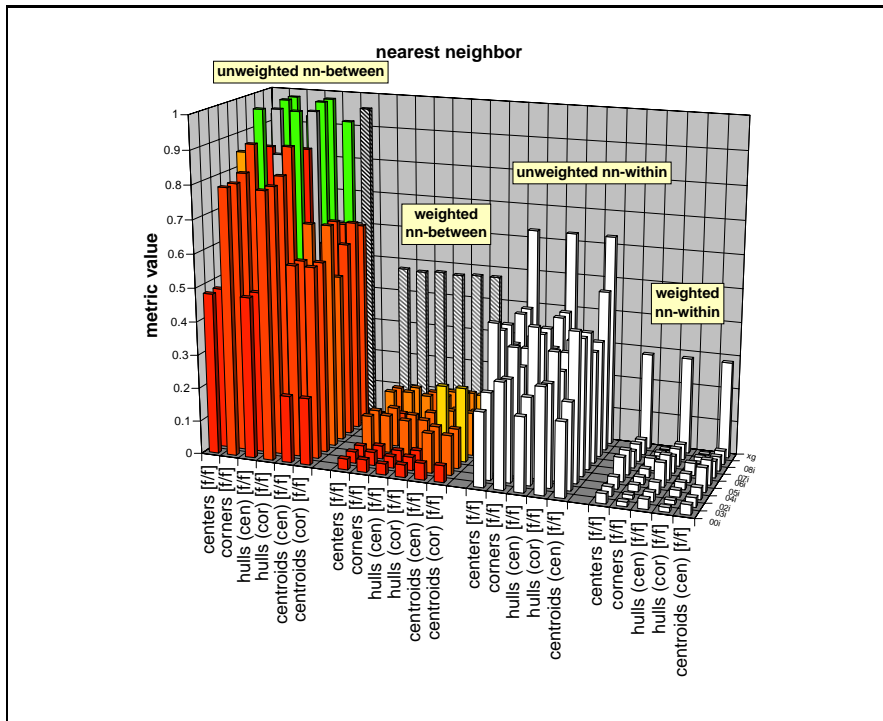Figure 9: Selected metric values for each drawing in stage 2.

Figure 9: Selected metric values for each drawing in stage 2. (continued)

the number of inversions: Two drawings D_i and D_j were judged to be out of order if D_i came after D_j in the metric ordering but before D_j in the human ordering, D_i and D_j were ranked equally in the metric ordering but not the human ordering, or D_i and D_j were ranked equally in the human ordering but not the metric ordering. The number of inversions was normalized by the maximum number of inversions for the sequence to adjust for different sequence lengths.

4.3 Experimental Results

Some of the metric names used in Figures 10 and 11 have modifiers:

- **nn-b** and **nn-w**: *unw* and *w* denote the unweighted and weighted versions, respectively
- **order**: *const* and *linear* denote the constant-weighted and linear-weighted versions, respectively
- **shape**: *norm* indicates that the normalized edit distance was used

4.3.1 Ordering Ability

Figure 10 shows the frequency with which different metric-and-pointset combinations did a particularly good or bad job of ordering the drawings for each graph. (Not all combinations of metrics and point sets shown in the figure were evaluted — the gray regions around the borders in Figure 10 mark combinations which were not computed.) A metric/pointset combination was flagged as doing a good job on a particular relaxation sequence if the number of inversions was noticeably lower than that for other metric/pointset combinations on the same sequence; similarly, a metric/pointset combination was flagged as doing a bad job if the number of inversions was noticeably higher than others. In a few cases no metric/pointset combination stood out as being noticeably better or worse than the others and so nothing was flagged for that sequence.

The orthogonal ordering metrics were strikingly better than most other metrics for all point sets except for those based on partition centroids. The point sets based on vertex corners fared particularly well, yielding the best ordering for over one-quarter of the sequences tested.

Shape, Euclidean distance, and the weighted nn-between metrics also stand out as being better than most of the other metrics, though they are not quite as good as orthogonal ordering.

Looking at the worst metrics, nn-within stands out as most often doing the worst job of ordering the drawings, particularly for point sets that are centroids of partitions of vertex corners. ϵ-clustering and partition size/count also do a consistently worse job of ordering.

In general, the partition centroid point sets were worse than the others, indicating that too much information is lost in these point sets to be of much use for
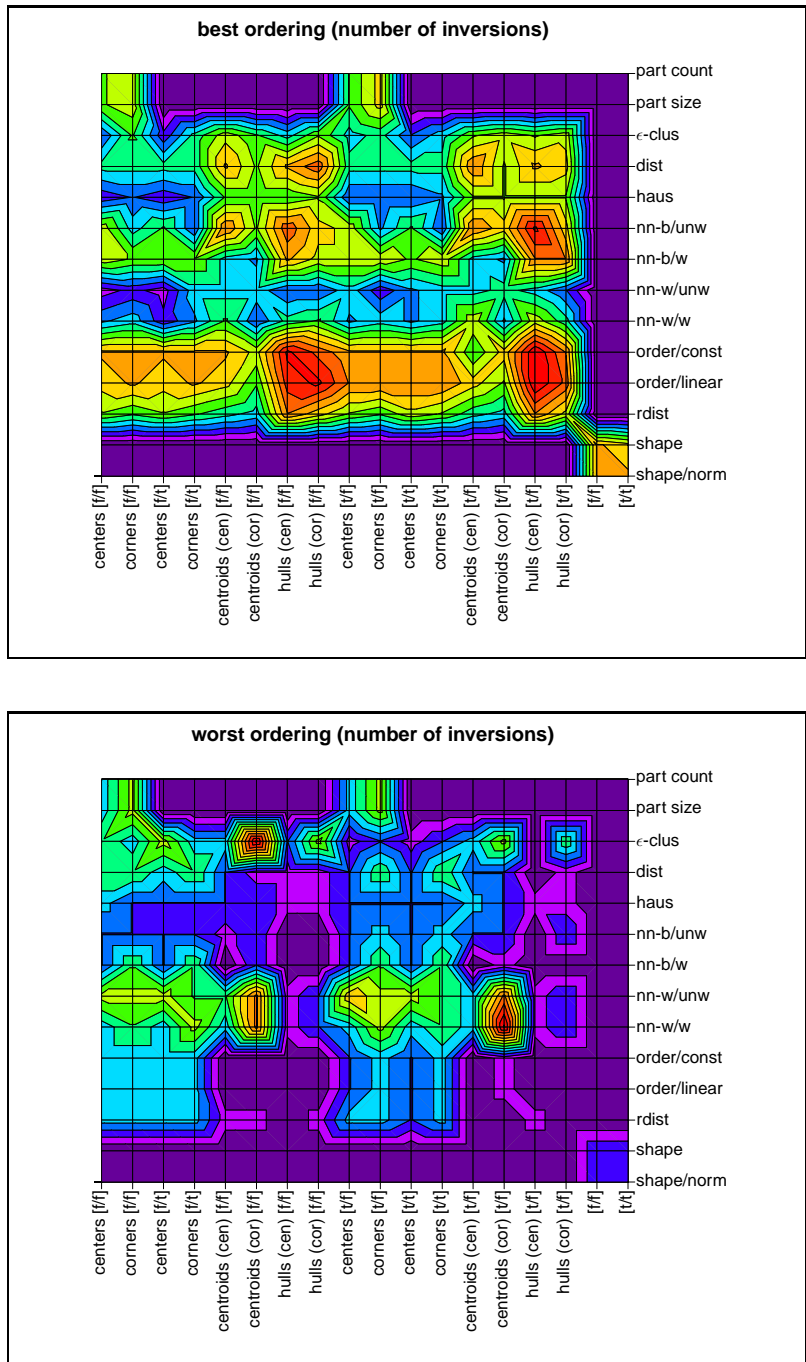
Figure 10: Purple is the lowest frequency; red is the highest. Peaks in the top picture indicate metric/pointset combinations which tend to produce orderings closest to the human-specified ordering; peaks in the bottom picture indicate combinations which tend to produce orderings most unlike the human-specified ordering.

similarity comparisons.

It is also important to consider the variation in the relative success of a given metric/pointset combination when applied to different graphs. Many did the worst job of ordering the drawings for at least one graph, and all did the best job at least once — even the most consistently best metric (orthogonal ordering) was the best only about 30% of the time. The variation is particularly large for the partition size and count metrics, as shown by a middle-of-the-road ranking in both the best and worst plots in Figure 10. These metrics are very sensitive to the slightest movement of vertices with respect to each other, and very quickly reach their maximum values if there is much change in the drawing.

The variation in performance of the metrics for different graphs can also be seen when breaking down the results by the type of modification. Shape, average relative distance, orthogonal ordering, nn-between, and the partition size and count metrics perform better on vertex deletions. Nearly all of the metrics, except nn-within and shape, perform noticeably worse on edge insertions. The partition size and count and the Hausdorff distance metrics also perform worse on vertex insertions. This behavior is explained by noting that in InteractiveGiotto, inserting an object into the graph tends to disrupt the drawing more than a deletion.

4.3.2   Rotation Ability

The other criterion evaluated at this stage is how suitable the metrics are for solving the rotation problem. The measurement for each metric/pointset combination is the percentage of drawings in the relaxation sequence for which the correct rotation was chosen. The percentage correct for each combination, averaged over all of the experimental graphs, is shown in Figure 11. (Note that nn-within is rotation-independent and is thus not included in the chart.)

The partition size and count metrics fared the worst, getting the correct rotation only 50-54% of the time. The low results for partition size and count are largely due to many cases where the metric could not distinguish between two or more rotations. This happened most often with the "more relaxed" drawings and reflects the fact that in these drawings the partitions tend to be very small.

The ϵ-clustering metric did somewhat better, getting 60-83% of the rotations correct; the large variation is due to whether the point set was based on vertex corners (worse) or vertex centers (better). The results here are due primarily to ϵ-clustering choosing the wrong rotation, rather than being unable to choose between multiple rotations. Point sets based on vertex corners are worse than those based on vertex centers because InteractiveGiotto places vertices so that the minimum spacing between vertices is the same as the minimum vertex dimension. As a result, the ϵ-cluster for each point typically does not contain more than four points — up to two other corners of the same vertex and up to two corners of neighboring vertices. This makes ϵ-clustering using vertex corners particularly sensitive to modifications which change vertex dimensions or separate vertices which are horizontally or vertically

near each other; however, if the vertices are spaced relatively far apart compared to the vertex size in both drawings, ϵ-clustering will report a small distance.

The unweighted nn-between metric also exhibited unsatisfactory behavior for some choices of point sets, getting 83-93% of the rotations correct. This is again the result of the metric being unable to distinguish between multiple rotations for the more relaxed drawings.

The average relative distance metric stands out as the best, averaging at least 93% of the rotations correct. This average is goes up to over 97% for the center and corner point sets.

Breaking down the results by the type of modification shows that there are again some variations between groups. The correct rotation was chosen most often when edge deletions were performed, followed by vertex insertions, edge splits, vertex deletions, and edge insertions (where the best metrics achieved only about 85% correctness). In most cases the relative performance of different metric/pointset combinations is the same as in the average of all the graphs, with a few notable exceptions:

- For graphs where an edge has been inserted, average relative distance with vertex corners and vertex centers is noticeably better and unweighted nn-between is noticeably worse than the other metrics.
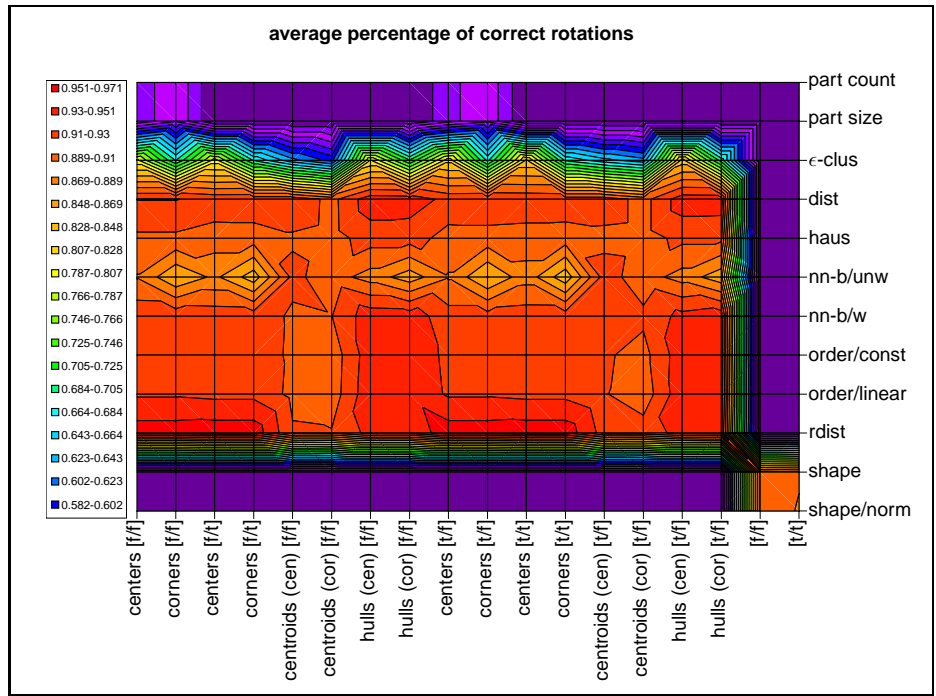

Figure 11: Purple is the lowest percentage correct (around 50%); red is the highest (94%).

- For graphs where an edge has been split, unweighted nn-between is noticeably worse for point sets consisting of vertex corners.

4.3.3   Conclusions

If the goal is simply to solve the rotation problem, average relative distance using either corner or center point sets performs quite well. Given the success of this metric, it seems unnecessary to consider anything more complicated to solve the rotation problem.

If it is important to compare different drawings of the same graph, the orthogonal ordering metrics are the best of the metrics tested in terms of qualitative behavior. The linear-weighted version has a slight edge over the constant-weighted one. Orthogonal ordering is also nearly as good as average relative distance in solving the rotation problem. However, there is still a good deal of room for improvement — orthogonal ordering only achieved the best ordering 30% of the time, and, considering all of the metrics, the correct ordering was found for only 9 of the 63 graphs.

5   Future Work

The next step is to study in more detail those cases for which certain metric/pointset combinations perform significantly worse (or better) than the average, since there are variations in performance between individual graphs. This may also indicate combinations of the existing metrics that may work better than any single one alone.

It will also be useful to test the metrics on drawings generated by other drawing algorithms. SMILE [2], for example, provides a way of obtaining many drawings of the same graph. Since the performance of some of the metrics can be traced to characteristics of InteractiveGiotto, this may prove illuminating.

Another important task is to analyze the quantitative behavior of the metrics. This requires that each drawing be assigned (by a human) a numerical value measuring how well the user's mental map is preserved. Obtaining these values is a non-trivial task, since it is difficult for a person to judge quantitatively the difference in visual distance between two pairs of drawings, even of the same graph — Is one pair twice as different as another? Only one-and-a-half times? Five percent less? Furthermore, asking if one drawing looks more like the original than another drawing may not be exactly the same question as asking which drawing does a better job of preserving the user's mental map. The chances are that a drawing which looks more like the original will do a better job of preserving the mental map, but assuming this presupposes something about how a user's mental map is structured. A solution to this may be to design an experiment in which the user gains familiarity with the original drawing, and is then timed on her response to a question involving a new drawing of the same graph. The idea is that a faster response time means

that the new drawing did a better job of preserving the mental map.

Other metrics can also be developed and evaluated. Metrics based on clustering and partitioning seem particularly related to how a user navigates around the drawing, and further work is needed to extend and improve these metrics. Surveying users about what they think makes two drawings more or less similar may also lead to additional metrics. Also, combinations of the current metrics may yield better results. Both current and new metrics should also be evaluated using a larger pool of graphs, including graphs of different sizes.

Finally, once suitable metrics have been identified and validated through user studies, they can be used to compare the behavior of interactive graph drawing algorithms as well as potentially providing inspiration for new drawing algorithms.

References

[1] H. Alt, O. Aichholzer, and G. Rote. Matching shapes with a reference point. *Internat. J. Comput. Geom. Appl.*, 1997. to appear.

[2] T. Biedl, J. Marks, K. Ryall, and S. Whitesides. Graph multidrawing: Finding nice drawings without defining nice. In S. Whitesides, editor, *Graph Drawing (Proc. GD '98)*, volume 1547 of *Lecture Notes Comput. Sci.*, pages 347–355. Springer-Verlag, 1998.

[3] T. C. Biedl and M. Kaufmann. Area-efficient static and incremental graph darwings. In R. Burkard and G. Woeginger, editors, *Algorithms (Proc. ESA '97)*, volume 1284 of *Lecture Notes Comput. Sci.*, pages 37–52. Springer-Verlag, 1997.

[4] U. Brandes and D. Wagner. A bayesian paradigm for dynamic graph layout. In G. Di Battista, editor, *Graph Drawing (Proc. GD '97)*, volume 1353 of *Lecture Notes Comput. Sci.*, pages 236–247. Springer-Verlag, 1998.

[5] U. Brandes and D. Wagner. Dynamic grid embedding with few bends and changes. In *Proceedings of the 9th Annual International Symposium on Algorithms and Computation (ISAAC'98)*, volume 1533 of *Lecture Notes in Computer Science*, pages 89–98. Springer-Verlag, 1998.

[6] S. S. Bridgeman, J. Fanto, A. Garg, R. Tamassia, and L. Vismara. InteractiveGiotto: An algorithm for interactive orthogonal graph drawing. In G. Di Battista, editor, *Graph Drawing (Proc. GD '97)*, volume 1353 of *Lecture Notes Comput. Sci.*, pages 303–308. Springer-Verlag, 1997.

[7] L. P. Chew, M. T. Goodrich, D. P. Huttenlocher, K. Kedem, J. M. Kleinberg, and D. Kravets. Geometric pattern matching under Euclidean motion. *Comput. Geom. Theory Appl.*, 7:113–124, 1997.

[8] R. F. Cohen, G. Di Battista, R. Tamassia, and I. G. Tollis. Dynamic graph drawings: Trees, series-parallel digraphs, and planar *ST*-digraphs. *SIAM J. Comput.*, 24(5):970–1001, 1995.

[9] G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimental comparison of three graph drawing algorithms. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 306–315, 1995.

[10] G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimental comparison of four graph drawing algorithms. *Comput. Geom. Theory Appl.*, 7:303–325, 1997.

[11] P. Eades, W. Lai, K. Misue, and K. Sugiyama. Preserving the mental map of a diagram. In *Proceedings of Compugraphics 91*, pages 24–33, 1991.

[12] U. Fößmeier. Interactive orthogonal graph drawing: Algorithms and bounds. In G. Di Battista, editor, *Graph Drawing (Proc. GD '97)*, volume 1353 of *Lecture Notes Comput. Sci.*, pages 111–123. Springer-Verlag, 1997.

[13] J. E. Goodman and R. Pollack. Multidimensional sorting. *SIAM J. Comput.*, 12(3):484–507, Aug. 1983.

[14] M. T. Goodrich, J. S. B. Mitchell, and M. W. Orletsky. Practical methods for approximate geometric pattern matching under rigid motion. *IEEE Trans. Pattern Anal. Mach. Intell.* to appear.

[15] K. Imai, S. Sumino, and H. Imai. Minimax geometric fitting of two corresponding sets of points. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 266–275, 1989.

[16] K. A. Lyons, H. Meijer, and D. Rappaport. Algorithms for cluster busting in anchored graph drawing. *J. Graph Algorithms Appl.*, 2(1):1–24, 1998.

[17] A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):926–932, Sept. 1993.

[18] K. Miriyala, S. W. Hornick, and R. Tamassia. An incremental approach to aesthetic graph layout. In *Proc. Internat. Workshop on Computer-Aided Software Engineering*, 1993.

[19] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *J. Visual Lang. Comput.*, 6(2):183–210, 1995.

[20] S. Moen. Drawing dynamic trees. *IEEE Softw.*, 7:21–28, 1990.

[21] S. North. Incremental layout in DynaDAG. In *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes Comput. Sci.*, pages 409–418. Springer-Verlag, 1996.

[22] A. Papakostas, J. M. Six, and I. G. Tollis. Experimental and theoretical results in interactive graph drawing. In S. North, editor, *Graph Drawing (Proc. GD '96)*, volume 1190 of *Lecture Notes Comput. Sci.*, pages 371–386. Springer-Verlag, 1997.

[23] A. Papakostas and I. G. Tollis. Interactive orthogonal graph drawing. In *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes Comput. Sci.* Springer-

Verlag, 1996.

[24] H. Purchase. Which aesthetic has the greatest effect on human understanding? In G. Di Battista, editor, *Graph Drawing (Proc. GD '97)*, volume 1353 of *Lecture Notes Comput. Sci.*, pages 248–261. Springer-Verlag, 1998.

[25] H. C. Purchase, R. F. Cohen, and M. James. Validating graph drawing aesthetics. In F. J. Brandenburg, editor, *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes Comput. Sci.*, pages 435–446. Springer-Verlag, 1996.

[26] K. Ryall, J. Marks, and S. Shieber. An interactive system for drawing graphs. In S. North, editor, *Graph Drawing (Proc. GD '96)*, volume 1190 of *Lecture Notes Comput. Sci.*, pages 387–393. Springer-Verlag, 1997.

[27] R. Tamassia, G. Di Battista, and C. Batini. Automatic graph drawing and readability of diagrams. *IEEE Trans. Syst. Man Cybern.*, SMC-18(1):61–79, 1988.