Shape-Faithful Graph Drawings

Amyra Meidiana, Seok-Hee Hong, and Peter Eades
University of Sydney, Australia
amei2916@uni.sydney.edu.au, {seokhee.hong, peter.eades}@sydney.edu.au

Abstract. Shape-based metrics measure how faithfully a drawing \( D \) represents the structure of a graph \( G \), using the proximity graph \( S \) of \( D \). While some limited graph classes admit proximity drawings (i.e., optimally shape-faithful drawings, where \( S = G \)), algorithms for shape-faithful drawings of general graphs have not been investigated.
In this paper, we present the first study for shape-faithful drawings of general graphs. First, we conduct extensive comparison experiments for popular graph layouts using the shape-based metrics, and examine the properties of highly shape-faithful drawings. Then, we present ShFR and ShSM, algorithms for shape-faithful drawings based on force-directed and stress-based algorithms, by introducing new proximity forces/stress.
Experiments show that ShFR and ShSM obtain significant improvement over FR (Fruchterman-Reingold) and SM (Stress Majorization), on average 12% and 35% respectively, on shape-based metrics.

1 Introduction

Recently, shape-based metrics [7] have been introduced for evaluating the quality of large graph drawing. It measures how faithfully the “shape” of a drawing \( D \) represents the ground truth structure of a graph \( G \), by comparing the similarity between the proximity graph \( S \) of the vertex point set of \( D \) and the graph \( G \).

For a point set \( P \) in the plane, proximity graphs are defined as: two points are connected by an edge if they are “close enough”. Specifically, a proximity region is defined for each pair of points, and if the proximity region is empty, the points are connected by an edge in the proximity graph [23].

Some limited graph classes always admit a proximity drawing \( D \), where the graph \( G \) is realized as a proximity graph \( S \) in \( D \). For such proximity drawable graph classes, some characterizations are known, and algorithms to construct such proximity drawings are available [13]. Consequently, such proximity drawings are optimally shape-faithful (i.e, shape-based metric of 1), since \( S = G \).

However, such optimally shape-faithful drawings are only applicable for very limited graph classes. Algorithms to optimize shape-based metrics for general graphs (i.e., not proximity drawable graphs) have not been studied yet.

In this paper, we present the first study for shape-faithful drawings of general graphs. Specifically, our main contributions can be summarized as follows:

* This work is supported by ARC grant DP190103301.
1. We evaluate the shape-faithfulness of popular graph drawing algorithms for various proximity drawable graph classes, including strong proximity drawable graphs (i.e., the best possible shape-based metric is 1), almost proximity drawable graphs with some forbidden subgraphs, weak proximity drawable graphs, and mesh graphs. Experiments show that tsNET \cite{10} obtains the highest shape-faithfulness on most large graph instances, for strong and almost proximity drawable graphs, and stress-based layouts \cite{11} achieve good results on mesh graphs.

2. We present ShFR and ShSM, algorithms for shape-faithful drawings for general graphs, based on the force-directed and stress-based layouts, by introducing new proximity forces/stress. Experiments with strong proximity drawable graphs, scale-free graphs and benchmark graphs show that ShFR and ShSM obtain significant improvement (on average, 12% and 35%) on the shape-based metrics over FR (Fruchterman-Reingold) \cite{9} and SM (Stress Majorization) \cite{11}.

2 Related Work

2.1 Shape-Based Metrics

Shape-based metrics measure how faithfully the “shape” of a drawing $D$ represents the ground truth structure of a graph $G$, by comparing the similarity between the proximity graph $S$ of the vertex point set of $D$ and the graph $G$ \cite{7}.

Specifically, the shape-based metrics use proximity graphs such as the Gabriel Graph ($GG$) and Relative Neighborhood Graph ($RNG$) (defined in Section 2.2). To compute the similarity between $G$ and $S$, both with vertex set $V$, the shape-based metrics use the Jaccard Similarity ($JS$) \cite{15} as follows: $JS(G, S) = \frac{1}{|V|} \sum_{v \in V} \frac{|N_G(v) \cap N_S(v)|}{|N_G(v) \cup N_S(v)|}$, where $N_G(v)$ (resp., $N_S(v)$) is the set of neighbors of vertex $v$ in $G$ (resp., $S$). We denote the shape-based metrics computed with this formula using $RNG$ (resp., $GG$) as $Q_{RNG}$ (resp., $Q_{GG}$), having values between 0 and 1 where 1 means perfectly shape-faithful.

2.2 Proximity Graphs

For a point set $P$ in the plane, a proximity graph $S$ of $P$ is roughly defined as follows: two points are connected by an edge if and only if they are “close enough”. Namely, the proximity region defined for the two points should be empty (i.e., contains no other points) \cite{23,24}. For example, Gabriel Graph ($GG$) \cite{10} (resp., Relative Neighborhood Graph ($RNG$) \cite{25}) is a proximity graph where two points $x$ and $y$ are connected by an edge if and only if the closed disk (resp., open lens) having line segment $xy$ as its diameter contains no other points.

For strong proximity, two conditions must be fulfilled: (a) two points are connected by an edge only if their proximity region is empty, and (b) two points are not connected by an edge only if their proximity region is not empty \cite{2}.

A relaxation of condition (b) gives rise to the definition of weak proximity, where the proximity graph may omit an edge between points $x$ and $y$ even if their
proximity region is empty [2]. Namely, while points need to be “close enough” to be connected by an edge in the proximity graph $S$, points can be made to be not connected by an edge in $S$ even if they are “close enough”.

2.3 Proximity Graph Drawing

Characterizations of strong proximity drawable graphs (i.e., graphs that admit a proximity drawing $D$, where the graph $G$ is realized as a proximity graph $S = G$ in $D$) are known for RNG and GG [3,17]:

- RNG-drawable graphs: trees with maximum degree 5, maximal outerplanar graphs, biconnected outerplanar graphs
- GG-drawable graphs: trees with maximum degree 4 and no degree 4 vertex with all “wide” subtrees, maximal and biconnected outerplanar graphs

Moreover, forbidden subgraphs have also been characterized: no GG- and RNG-drawable graphs may contain $K_4$ and $K_{2,3}$ as subgraphs [10].

Characterizations of weak proximity drawable graphs include wider classes:

- trees (regardless of maximum degree): weak GG- and RNG-drawable [2]
- 1-connected outerplanar graphs with no vertex of degree 1: weak GG-drawable [8].

Algorithms to construct proximity drawings of both strong and weak proximity drawable graphs are available [2,3,17], although implementations are unavailable and challenging due to requiring precise geometric computations. For details on proximity graph drawing, see a survey [19].

3 Graph Layout Comparison Experiments

3.1 Experiment Design and Data Sets

In this Section, we present extensive experiments using the shape-based metrics $Q_{RNG}$ and $Q_{GG}$ to compare popular graph drawing algorithms:

- Force-directed layouts: Fruchterman-Reingold (FR) [9], Organic (OR) [27].
- Multi-level force-directed layouts: $FM^3$ [12], $sfdp$ [14].
- Backbone layout (BB) [22], which untangles hairballs in a drawing.
- LinLog layout (LL) [21], a force-directed algorithm displaying clusters.
- Stress-based layouts to minimize the stress: Stress Majorization (SM) [11], Stochastic Gradient Descent (SGD) [28].
- $tsNET$ layout [16], based on the t-SNE dimension reduction [20].
- Walker’s level drawing algorithm (W) for trees [26].
- Chrobak and Kant algorithm (CK) [5] for convex grid drawings of triconnected planar graphs in quadratic area.
For data sets, we generate graphs with various sizes: small graphs with 50-250 vertices, medium graphs with 250-500 vertices, and large graphs with 500-1000 vertices. Furthermore, we consider graph types based on proximity drawability characterization: strong proximity drawable graphs, almost proximity drawable graphs, and weak proximity drawable graphs. We also use mesh graphs, which do not fall into known proximity drawability characterizations. For each graph type and size, we generate ten graph instances.

**Strong Proximity Drawable Graphs:** We generate strong proximity drawable graphs based on known characterizations [3,17]:

- **Maximum outerplanar graphs**, generated using the connected planar graph generator of OGDF [4].
- **Biconnected outerplanar graphs:** We start $G$ as a cycle of random length $\leq$ the target size $n$. Then, select an edge $(u, v)$ in $G$ that is only involved in one cycle. Select a cycle length $x < n$, create a path $p$ of length $x - 2$, and add an edge between $u$ and the first vertex of $p$, and between $v$ and the last vertex of $p$. Repeat while the number of vertices in $G$ is less than $n$.
- **Proximity drawable trees**, generated using the random tree generator of OGDF: For RNG-drawable trees, we set the maximum vertex degree as 5; for GG-drawable trees, we set the maximum vertex degree as 4, and then prune forbidden subtrees until the tree contains no more forbidden subtrees.

**Almost Proximity Drawable Graphs with Forbidden Subgraphs:** We start with a strong proximity drawable graph $G$, and then add a few edges and/or vertices to create a forbidden subgraph. The number of edges (resp., vertices) added are limited to at most 10 (resp., 5). Specifically, we perform two types of forbidden subgraph augmentation:

- **L-AUG (Local Augmentation)** graphs: We choose a vertex $v$ of $G$ and add new vertices and edges around $v$ to create a forbidden subgraph $F$.
- **F-AUG (Global Augmentation)** graphs: We select a subset of vertices of $G$, all separated by a shortest path length above a predefined threshold, and add edges between the selected vertices to create a forbidden subgraph $F$.

**Weak Proximity Drawable Graphs:** We also use weak proximity drawable graphs based on the weak proximity drawability characterization [2]:

- **1-connected** outerplanar graph with a minimum degree of 2, which are weak GG-drawable [5]: We generate the graphs in a similar way to the biconnected outerplanar graphs, however alternately appending the new cycle to a random vertex rather than a random edge.

**Mesh Graphs:** We use simple mesh graphs containing no chordless cycles of length $> 3$, from the jagmesh set of the SuiteSparse Matrix collection [6]. These graphs are not part of known proximity drawability characterizations, but can be drawn as an RNG drawing, by drawing each 3-cycle as an equilateral triangle.
3.2 Results

**Strong Proximity Drawable Graphs** On strong proximity drawable trees, all the drawing algorithms used fail to obtain shape-based metrics close to optimal.

Figure 1 shows the average $Q_{RNG}$ for RNG-drawable trees. On small trees, the best performing layouts, OR, BB, multi-level layouts, and stress-based layouts, only obtain $Q_{RNG}$ of 0.5-0.6 on average. $tsNET$ becomes the best performing layout on medium and large trees, with $Q_{RNG}$ of about 0.4 on average. On large trees, the differences in $Q_{RNG}$ between layouts are more pronounced, with $tsNET$ and $LL$ performing the best, followed by $sfdp$ and OR.

For small proximity drawable outerplanar graphs (both GG- and RNG-drawable), the best performing layouts, stress-based layouts and BB, obtain $Q_{RNG}$ of around 0.7 (see Figure 2). This is notably closer to optimal compared to RNG-drawable trees, where all layouts obtain average $Q_{RNG}$ of at most 0.6. $tsNET$ and $CK$ are the top performing layouts on medium and large outerplanar graphs, despite lower performance on small graphs: on medium and large maximum outerplanar graphs, $tsNET$ (resp., $CK$) obtains $Q_{RNG}$ of 0.6 (resp., 0.5) on average. This is closer to optimal compared to large RNG-drawable trees, where $tsNET$, the best performing layout, only obtains average $Q_{RNG}$ of 0.4.

For GG-drawable trees, the results on $Q_{GG}$ are mostly similar to $Q_{RNG}$: similarly, for GG-drawable outerplanar graphs (same set of graphs as RNG-drawable outerplanar graphs), the results on $Q_{GG}$ are similar to $Q_{RNG}$. For details, see Figures 7 and 8 in Appendix B.
Table 1. Example layout comparison for a large RNG-drawable tree.

| BB | FM° | FR | LL | OR |
|----|-----|----|----|----|
| ![Image](image1.png) | ![Image](image2.png) | ![Image](image3.png) | ![Image](image4.png) | ![Image](image5.png) |
| ![Image](image6.png) | ![Image](image7.png) | ![Image](image8.png) | ![Image](image9.png) | ![Image](image10.png) |

Table 2. Example layout comparison for a large maximum outerplanar graph.

| BB | FM° | FR | LL | OR |
|----|-----|----|----|----|
| ![Image](image11.png) | ![Image](image12.png) | ![Image](image13.png) | ![Image](image14.png) | ![Image](image15.png) |
| ![Image](image16.png) | ![Image](image17.png) | ![Image](image18.png) | ![Image](image19.png) | ![Image](image20.png) |

Table 1 shows a visual comparison of graph layouts on a large RNG-drawable tree. For the best performing layouts tsNET and LL, subtrees closer to the leaves are often “compacted” together, compared to the second best performing layouts such as OR and sfdp, where all branches are more “opened” up.

Table 2 shows a visual comparison of layouts on a maximal outerplanar graph. The best performing layout, tsNET, collapses the faces on the periphery, compared to the faces in the middle of the drawing. The “long” drawing of CK may have obtained a comparable effect, producing high shape-based metrics.
Almost Proximity Drawable Graphs In general, the ranking of the graph drawing algorithms on the shape-based metrics do not change much between strong proximity drawable graphs and almost proximity drawable graphs.

Figure 3 shows comparisons on $Q_{RNG}$ for the base RNG drawable trees and the L-AUG and F-AUG graphs, where tsNET still obtains the highest $Q_{RNG}$. LL also obtains the second highest $Q_{RNG}$, although with a smaller difference to the next best performing layouts OR and sfdp, compared to RNG drawable trees.

Table 3 shows a visual comparison on a F-AUG graph, where the layouts with highest shape-based metrics, such as tsNET and LL, draw the “branches” in the periphery of the drawing in a more compact way, than other layout. This observation is consistent with the pattern also seen in the visual comparison for strong proximity drawable trees and outerplanar graphs.
Fig. 4. Average $Q_{GG}$ for 1-connected outerplanar graphs. OR and CK performs the best on large 1-connected outerplanar graphs.

Table 4. Example layout comparison for a large 1-connected outerplanar graph.

| BB   | FM$'$ | FR   | LL   | OR   |
|------|-------|------|------|------|
| sfdp | SGD   | SM   | tsNET| CK   |

Weak Proximity Drawable Graphs For weak $GG$-drawable 1-connected outerplanar graphs, OR surprisingly obtains the highest $Q_{GG}$ on large 1-connected outerplanar graphs, followed by CK and tsNET; see Figure 4.

Table 4 shows a visual comparison, where OR draws a number of chordless cycles with their vertices in a regular polygon configuration. In fact, this is the correct way to draw such cycles as $GG$, resulting in high $Q_{GG}$.

Mesh Graphs On mesh graphs, the best performing layouts, stress-based layouts, obtain on average much higher shape-based metrics than on other strong proximity drawable graphs, see Figure 5. In particular, SGD and SM obtain near-perfect shape-based metrics ($Q_{RNG} = 0.99$ on average), and OR and BB also obtain very high shape-based metrics ($Q_{RNG} = 0.98$ on average). On the other hand, tsNET obtains comparatively lower shape-based metrics.

Table 5 shows a visual comparison on a mesh graph; most layouts manage to untangle the mesh. Furthermore, SGD and SM manage to untangle without twists or “distortions”, where triangles in the periphery are more “squashed” compared to the triangles in the middle, as seen in sfdp or tsNET layouts.
Fig. 5. Average $Q_{RNG}$ for mesh graphs. Stress-based layouts obtain the best shape-based metrics, at almost perfect.

Table 5. Example layout comparison for mesh.

|   | BB | FM$^3$ | FR | LL | OR |
|---|----|--------|----|----|----|
| sfdp | SGD | SM | tsNET | CK |

3.3 Discussion and Summary

Overall, tsNET performs the best on large strong proximity drawable graphs, followed by LL. Looking at the visual comparison, these layouts often “collapse” subgraphs on the periphery. This may have lead to fewer non-adjacent vertices being close to each other, leading to better shape-based metrics. Moreover, this improvement compared to other layouts is more apparent in larger graphs, where the larger number of vertices means more non-adjacent vertices being close to each other in drawings where subgraphs on the periphery are not “collapsed”.

Most layout algorithms are better at computing drawings closer to optimal shape-faithfulness for dense strong proximity drawable graphs: the best-performing layouts, tsNET and LL, obtain much higher average shape-based metrics on outerplanar graphs compared to trees. Lower density means more pairs of vertices are not adjacent in $G$, i.e., more proximity regions need to be non-empty in $D$.

The mesh graphs are drawn as RNG by drawing each face as equilateral triangles, i.e., having uniform edge lengths, a readability metric which is often used as a goal for a number of layout algorithms. This may be why more layout
algorithms, especially stress-based layouts which emphasize distance faithfulness, are able to produce almost-perfect shape-faithful drawings for the mesh graphs.

4 Algorithms for Shape-Faithful Graph Drawings

In this Section, we present algorithms for shape-faithful drawings. Based on the qualitative observations from the layout comparison experiments in Section 3, high shape-based metrics are obtained often by “collapsing” subgraphs on the drawing’s periphery - this keeps non-adjacent vertices in $G$ distant from each other, and adjacent vertices in the collapsed subgraphs within close proximity. Therefore, our main idea for shape-faithful graph drawings is to “drive away” non-adjacent vertices in $G$ that are geometrically too close in the drawing $D$.

Specifically, we present two algorithms $ShFR$ and $ShSM$ based on two popular graph drawing algorithms, force-directed and stress minimization algorithms. $ShFR$ and $ShSM$ aim to improve shape-based metrics by introducing two new types of proximity forces/stress. For a pair of adjacent vertices $v$ and $u$ in $G$ and another vertex of $t$ currently located in the proximity region of $v$ and $u$ in $D$:

- proximity repulsion force/stress: push $t$ out of the proximity region of $u$, $v$;
- proximity attraction force/stress: pull $v$ and $u$ closer together.

4.1 $ShFR$: Force-Directed Layout for Shape-Faithful Drawings

We present $ShFR$, a force-directed layout for shape-faithful drawing, incorporating proximity forces with Fruchterman-Reingold ($FR$) [9].

To explain the design rationale for $ShFR$, consider the following case: for a pair of adjacent vertices $u$ and $v$ in a graph $G = (V, E)$, the edge $(u, v)$ does not exist in the proximity graph $S = (V, E')$ of a drawing $D$ of $G$, due to a vertex $t$ located inside the proximity region of $u$ and $v$ in $D$. For such a case, to add back the edge $(u, v)$ in the proximity graph $S$ to achieve $S = G$, we introduce two new proximity forces: (1) repulsion force to repel $t$ out of the proximity region of $u$ and $v$; (2) attraction force on $u$ and $v$ to shrink the proximity region.

We first add a proximity repulsion force to drive $t$ out of the proximity region of $u$ and $v$ in $D$. From the midpoint $m$ between $u$ and $v$, we add a repulsion force acting on $t$, with a magnitude proportional to how far $t$ needs to be away from $m$ in order to be driven out of the proximity region of $u$ and $v$. Specifically, the $x$-displacement of $t$ induced by the repulsion force can be computed as:

$$x_t - x_m = f ||X_u - X_v|| l^2 ||X_t - X_m||$$

where $x_t$ is the $x$-coordinate of $t$, $||X_t - X_m||$ is the Euclidean distance between $t$ and $m$, $l$ is the parameter for ideal spring length (i.e., target edge length), and $f$ is the parameter for spring stiffness.

Next, we add a proximity attraction force for a pair of adjacent vertices $u$ and $v$ in $G$ with non-empty proximity regions. Specifically, we add an attraction force acting between $u$ and $v$: $(x_u - x_v)(||X_u - X_v||)l^{-1}$.

The new proximity forces can be added to any force-directed algorithms. For our specific implementation, we add the proximity forces in conjunction with $FR$,
where the proximity force computations are added to each force computation iteration of FR. For details, see Algorithm 1 in Appendix C.

GG and RNG are subgraphs of the Delaunay Triangulation, which can be computed in $O(n \log n)$ time [23]. The original FR algorithm runs in $O(n^2)$ time. Therefore, the total runtime of ShFR is $O(n^2)$.

4.2 ShSM: Stress-Based Layout for Shape-Faithful Drawings

We now present ShSM for shape-faithful drawing, incorporating proximity stress with Stress Majorization (SM) [11]. Similar to the force-directed case, for each case where in drawing $D$ a vertex $t$ lies in the proximity region of two neighboring vertices $v$ and $u$, i.e. $(u, v) \in E$ but $(u, v) \notin E'$, we add two new types of stress: (1) repulsion stress to push $t$ out of the proximity region; (2) attraction stress to pull $v$ and $u$ closer together.

We first add the proximity repulsion stress by exerting stress on $t$ from the midpoint $m$ of $u$ and $v$. Specifically, we compute the $x$-displacement of $t$ due to the stress between $t$ and $m$ as $w_{uv}^r(x_m) + d_{uv}(x_m - x_t) ||X_v - X_u||/||X_t - X_m||$, where $d_{uv}$ is the shortest path distance between $u$ and $v$ and $w_{uv}$ is the weight computed for the vertex pair $u$ and $v$, often computed as $(d_{uv})^k$ for a constant $k$. Since $m$ is not an actual vertex of $G$, there is no graph theoretic distance or weight between $m$ and $t$; we instead use $d_{uv}$ and $w_{uv}$, and then scale them based on the ratio of the Euclidean distances between $u, v$, and between $t, m$.

We next add the proximity attraction stress which has a weight lower than the standard stress of SM, to attract $u$ and $v$ closer in order to reduce the distance between $u$ and $v$. The $x$-displacement of $v$ due to this additional stress is computed as $w_{uv}^a(x_u) + d_{uv}(x_v - x_u)/||X_v - X_u||$, where $w_{uv}^a = w_{uv}^a k'$ for $k' < 1$.

The new proximity stress can be added to any stress-based algorithms. For our specific implementation, we add the proximity stress in conjunction with SM, where the proximity stress computations are added to each stress computation iteration of SM. For details, see Algorithm 2 in Appendix D.

As with ShFR, GG and RNG can be computed in $O(n \log n)$ time and the original stress computation of SM takes $O(n^2)$ time. The total runtime of ShSM is therefore $O(n^2)$.

5 ShFR and ShSM Experiments

5.1 Experiment Design and Data Sets

In this experiment, we evaluate the effectiveness of ShFR and ShSM over FR and SM respectively, using shape-based metrics $Q_{RNG}$ and $Q_{GG}$.

For data sets, we use strong proximity drawable graphs, as well as scale-free graphs and benchmark graphs:

- strong proximity drawable graphs, from Section 3
- scale-free graphs: We generate synthetic scale-free graphs with density 2, 3, and 5, using the NetworkX [13] scale-free generator.
– benchmark graphs, including real-world scale-free graphs [6,18,29] with up to 6000 vertices and 15000 edges. For details, see Table 8 in Appendix A.

To measure the improvement of the shape-based metrics, for example, on $Q_{RNG}$ by $ShFR$ over $FR$, we define the formula $I(Q_{RNG}) = \frac{Q_{RNG}(ShFR) - Q_{RNG}(FR)}{Q_{RNG}(FR)}$. We use the same formula for $Q_{GG}$, and for the improvement by $ShSM$ over $SM$.

5.2 Results

$ShFR$ obtains notable improvement over $FR$ on $Q_{RNG}$ and $Q_{GG}$ for large strong proximity drawable graphs, obtaining average improvement of 15%, 12%, and 12% on maximum outerplanar graphs, biconnected outerplanar graphs, and trees respectively, see Figure 6 (a). $ShFR$ also obtains significant improvement over $FR$ on $Q_{GG}$ for scale-free graphs, at on average 18%. For real-world benchmark graphs, the improvement on $Q_{RNG}$ and $Q_{GG}$ average at around 10%.

$ShSM$ obtains significant improvement over $SM$ for strong proximity drawable graphs, see Figure 6 (b). For maximum outerplanar graphs, $ShSM$ obtains significant improvement over $SM$ (average 20% and 25%) on $Q_{RNG}$ and $Q_{GG}$ respectively, which is much higher than the improvement by $ShFR$ over $FR$. For biconnected outerplanar graphs, an even larger improvement of on average 40% is achieved on $Q_{GG}$. For large trees, $ShSM$ also obtains significant improvement over $SM$, on average 18% and 30% on $Q_{RNG}$ and $Q_{GG}$, respectively.

$ShSM$ also obtains significant improvement over $SM$ for scale-free graphs, on average 20% improvement on $Q_{RNG}$. Notably, the largest improvement is obtained by $ShSM$ on $Q_{GG}$ for scale-free graphs, at over 70%, on scale-free graphs. Note that $ShSM$ obtains on average 20% and 42% improvement over $SM$ for real-world benchmark graphs, on $Q_{RNG}$ and $Q_{GG}$ respectively.

Table 6 shows a visual comparison of $FR$ and $ShFR$ on the benchmark scale-free graph $G_A$. $ShFR$ untangles the “hairball” more clearly, compared to
Table 6. Visual comparison of FR and ShFR, SM and ShSM on benchmark graphs. ShFR often untangles the hairballs better than FR, and ShSM expands faces that are “collapsed” by SM.

| FR | ShFR |
|----|------|
| ![Image](image1.png) | ![Image](image2.png) |

| SM | ShSM |
|----|------|
| ![Image](image3.png) | ![Image](image4.png) |

FR. Table 6 also shows a visual comparison of SM and ShSM on the benchmark scale-free graph netscience. ShSM “expands” faces that are “squashed” in SM, showing the local neighborhood of some vertices more clearly. However, the expanded faces also leads to the drawing feeling more “crowded” compared to SM, thus increasing faithfulness but affecting readability. For more visual comparisons on other data sets, see Table 9 in Appendix E.

5.3 Discussion and Summary

Our extensive experiments demonstrate the effectiveness of ShFR and ShSM for shape-faithful drawings. ShFR (resp., ShSM) obtains significant improvement over FR (resp., SM) of 11% and 13% (resp., 20% and 50%) on Q_RNG and Q_GG respectively, averaged over all data sets.

For strong proximity drawable graphs, ShFR (resp., ShSM) obtains improvement over FR (resp., SM) of on average 13% and 13% (resp., 20% and 30%) on Q_RNG and Q_GG respectively. For real-world benchmark graphs, ShFR (resp., ShSM) obtains improvement over FR (resp., SM) of on average 10% and 10% (resp., 20% and 43%) on Q_RNG and Q_GG respectively. For scale-free
graphs, \textit{ShFR} (resp., \textit{ShSM}) obtains improvement over \textit{FR} (resp., \textit{SM}) of on average 10\% and 16\% (resp., 17\% and 70\%) on \textit{Q_{RNG}} and \textit{Q_{GG}} respectively. Notably, the \textit{Q_{GG}} improvement of \textit{ShSM} over \textit{SM} on scale-free graphs at 70\% is the largest among all data sets.

The improvements are much higher for large graphs. In general, large graphs have many vertex pairs, with a high ratio of non-adjacent vertices to adjacent pairs of vertices in \textit{G}. Therefore, there are potentially more vertices located in proximity region that should be empty, creating more instances for the proximity forces and stress to improve the shape-based metrics.

Furthermore, the best improvement is achieved by \textit{ShSM} over \textit{SM} on \textit{Q_{GG}}, significantly higher than the improvement on \textit{Q_{RNG}} and the improvements of \textit{ShFR} over \textit{FR}. Specifically, larger improvements are obtained on \textit{Q_{GG}} than \textit{Q_{RNG}} on scale-free and real-world benchmark graphs by \textit{ShSM}. Since the proximity region of \textit{RNG} (i.e., lens at points \textit{u} and \textit{v}) is larger than the proximity region of \textit{GG} (i.e., disk with \textit{uv} as diameter), when applying proximity stress, it is harder to push all non-adjacent vertices out of the proximity region of \textit{RNG}. In addition, the tendency for \textit{ShSM} to “open up” collapsed faces compared to \textit{ShFR} may have led to the better improvements obtained by \textit{ShSM}.

6 Conclusion and Future Work

In this paper, we present the first study for the shape-faithful drawings of general graphs. We first evaluate the shape-faithfulness of existing graph layouts and examine the properties of good shape-faithful drawings. In general, \textit{tsNET} obtains the highest shape-faithfulness on medium-to-large graphs.

We then present \textit{ShFR} and \textit{ShSM}, algorithms for shape-faithful drawings of general graphs, based on force-directed and stress-based layouts, introducing new proximity forces/stress. Extensive experiments show that \textit{ShFR} and \textit{ShSM} achieve significant improvement over \textit{FR} and \textit{SM}, on average, 12\% and 35\% higher shape-based metrics respectively. Notably, \textit{ShSM} obtains a 70\% average improvement on \textit{Q_{GG}} over \textit{SM} for scale-free graphs.

Future work includes shape-faithful layouts based on various other layouts.
References

1. Battista, G.D., Lenhart, W., Liotta, G.: Proximity drawability: A survey extended abstract. In: International Symposium on Graph Drawing. pp. 328–339. Springer (1994)
2. Battista, G.D., Liotta, G., Whitesides, S.: The strength of weak proximity. In: International Symposium on Graph Drawing. pp. 178–189. Springer (1995)
3. Bose, P., Lenhart, W., Liotta, G.: Characterizing proximity trees. Algorithmica 16(1), 83–110 (1996)
4. Chimani, M., Gutwenger, C., Jünger, M., Klau, G.W., Klein, K., Mutzel, P.: The open graph drawing framework (ogdf). Handbook of Graph Drawing and Visualization 2011, 543–69 (2013)
5. Chrobak, M., Kant, G.: Convex grid drawings of 3-connected planar graphs. International Journal of Computational Geometry & Applications 7(03), 211–223 (1997)
6. Davis, T.A., Hu, Y.: The university of florida sparse matrix collection. ACM Transactions on Mathematical Software (TOMS) 38(1), 1 (2011)
7. Eades, P., Hong, S.H., Nguyen, A., Klein, K.: Shape-based quality metrics for large graph visualization. J. Graph Algorithms Appl. 21(1), 29–53 (2017)
8. Evans, W., Gansner, E.R., Kaufmann, M., Liotta, G., Meijer, H., Spillner, A.: Approximate proximity drawings. In: International Symposium on Graph Drawing. pp. 166–178. Springer (2011)
9. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Software: Practice and Experience 21(11), 1129–1164 (1991)
10. Gabriel, K.R., Sokal, R.R.: A new statistical approach to geographic variation analysis. Systematic zoology 18(3), 259–278 (1969)
11. Gansner, E.R., Koren, Y., North, S.: Graph drawing by stress majorization. In: International Symposium on Graph Drawing. pp. 239–250. Springer Berlin Heidelberg (2004)
12. Hachul, S., Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: International Symposium on Graph Drawing. pp. 285–295. Springer Berlin Heidelberg (2004)
13. Hagberg, A., Swart, P., Chult, D.S.: Exploring network structure, dynamics, and function using networkx. Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (1 2008)
14. Hu, Y.: Efficient, high-quality force-directed graph drawing. Mathematica Journal 10(1), 37–71 (2005)
15. Jaccard, P.: The distribution of the flora in the alpine zone. 1. New Phytologist 11(2), 37–50 (1912)
16. Krueger, J.F., Rauber, P.E., Martins, R.M., Kerren, A., Kobourov, S., Telea, A.C.: Graph layouts by t-sne. Computer Graphics Forum 36(3), 283–294 (2017). https://doi.org/10.1111/cgf.13187
17. Lenhart, W., Liotta, G.: Proximity drawings of outerplanar graphs. In: International Symposium on Graph Drawing. pp. 286–302. Springer (1996)
18. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data (Jun 2014)
19. Liotta, G.: Proximity Drawings, chap. 4, pp. 115–154 (2013)
20. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research 9(86), 2579–2605 (2008)
21. Noack, A.: An energy model for visual graph clustering. In: International symposium on graph drawing. pp. 425–436. Springer Berlin Heidelberg (2003)
22. Nocaj, A., Ortmann, M., Brandes, U.: Untangling the hairballs of multi-centered, small-world online social media networks. Journal of Graph Algorithms and Applications 19(2), 595–618 (2015). [https://doi.org/10.7155/jgaa.00370]
23. Preparata, F.P., Shamos, M.I.: Computational geometry: an introduction. Springer Science & Business Media (2012)
24. Toth, C.D., O’Rourke, J., Goodman, J.E.: Handbook of discrete and computational geometry. CRC press (2017)
25. Toussaint, G.T.: The relative neighbourhood graph of a finite planar set. Pattern recognition 12(4), 261–268 (1980)
26. Walker, J.Q.: A node-positioning algorithm for general trees. Software: Practice and Experience 20(7), 685–705 (1990)
27. Wiese, R., Eiglsperger, M., Kaufmann, M.: yfiles - visualization and automatic layout of graphs, pp. 173–191. Springer Berlin Heidelberg (2004). [https://doi.org/10.1007/978-3-642-18638-7_8]
28. Zheng, J.X., Pawar, S., Goodman, D.F.: Graph drawing by stochastic gradient descent. IEEE transactions on visualization and computer graphics 25(9), 2738–2748 (2018)
29. Zitnik, M., Sosić, R., Maheshwari, S., Leskovec, J.: BioSNAP Datasets: Stanford biomedical network dataset collection. [http://snap.stanford.edu/biodata](Aug 2018)
Appendix A  Data Sets

Table 7. Data set details for layout comparison experiments. Note that while the mesh graphs we use do not fall under known proximity drawability characterizations, they can be drawn as RNG by drawing each 3-cycle as an equilateral triangle.

| Name                   | RNG-drawable | GG-drawable | Weak GG-drawable | Avg. density |
|------------------------|--------------|-------------|------------------|--------------|
| Tree (max deg. 5)      | Y            | N           | Y                | 0.99         |
| Tree (non-forbidden max deg. 4) | Y            | Y           | Y                | 0.99         |
| Max. outerplanar       | Y            | Y           | Y                | 2.01         |
| Biconn. outerplanar    | Y            | Y           | Y                | 1.40         |
| 1-conn. outerplanar    | N            | N           | Y                | 1.50         |
| L-AUG                  | N            | N           | N                | 1.01         |
| F-AUG                  | N            | N           | N                | 1.01         |
| Mesh                   | N*           | N           | N                | 2.85         |

Table 8. Benchmark data set details for ShFR and ShSM experiments

| Name            | | | density |
|-----------------|---|---|---------|
| as19990606      | 5188 | 9930 | 1.91    |
| migrations_lcc  | 6025 | 9378 | 1.56    |
| netscience      | 379  | 914  | 2.41    |
| oflights_lcc    | 2905 | 15645| 5.39    |
| tvcg            | 3213 | 10140| 3.16    |
| us_powergrid    | 4941 | 6594 | 1.33    |
| yeastppsi_lcc   | 2224 | 6609 | 2.97    |
| 1138_bus        | 1138 | 2596 | 2.28    |
| add32           | 4960 | 14422| 2.91    |
| eva             | 4475 | 4652 | 1.04    |
| G_4             | 2075 | 4769 | 2.30    |
Appendix B  Graph Layout Comparison Results

Fig. 7. Average $Q_{GG}$ for strong GG-drawable trees. The pattern in best-performing drawing algorithms are similar with RNG-drawable trees on medium and large trees. Even highest-performing layouts are still far from ideal ($Q_{GG} = 1$).

Fig. 8. Average $Q_{GG}$ for maximum outerplanar graphs. The ordering between layouts is mostly the same as with $Q_{RNG}$, with tsNET and CK obtaining the best $Q_{GG}$ on medium and large graphs.
Appendix C  \textit{ShFR} Algorithm

Before explaining our algorithm, we first provide an explanation of force-directed methods. Force-directed algorithms model a graph as a system of bodies with two types of forces acting between them: a \textit{repulsion} force for each pair of vertices, and an \textit{attraction} force for each edge.

For two vertices \textit{u}, \textit{v}, the \textit{x-displacement} \( x_v' \) of \( v \) induced by the repulsion force exerted by \( u \) on \( v \) can be computed as
\[
\frac{x_v - x_u}{||X_v - X_u||^2} \times l^2,
\]
where \( x_v \) is the \textit{x}-coordinate of \( v \), \( ||X_v - X_u|| \) is the Euclidean distance between \( u \) and \( v \), \( l \) is a parameter representing natural spring length (i.e., the target edge length), and \( f \) is a parameter for spring stiffness.

For two adjacent vertices \textit{u}, \textit{v} (i.e., edge \((u, v)\) in \( G \)), the \textit{x-displacement} of \( v \) induced by the attraction force exerted by \( u \) on \( v \) can be computed as
\[
(x_u - x_v)(||X_v - X_u||)l^{-1}.
\]

Algorithm 1 describes the details of \textit{ShFR} using the pseudo code.

Appendix D  \textit{ShSM} Algorithm

Before explaining our algorithm, we first explain stress-based algorithms using Stress Majorization (\textit{SM}), a popular stress-based algorithm. Stress-based algorithms aim to minimize the stress in a drawing, where low stress means that the Euclidean distances in the drawing are proportional to the graph-theoretic distances in a graph.

For two vertices \textit{v} and \textit{u}, the \textit{stress} between the two vertices is defined by how proportional the Euclidean distances between the two vertices in \( D \) is to the length of the shortest path between the two vertices in \( G \). More precisely, the \textit{x-displacement} \( x_v' \) of \( v \) induced by the stress between \( v \) and \( u \) can be computed as
\[
w_{uv}(x_v) + d_{uv}(x_v - x_u)/(||X_v - X_u||),
\]
where \( d_{uv} \) is the graph-theoretic distance between \( v \) and \( u \), and \( w_{uv} \) is a weight for the pair \( u \) and \( v \), defined as \((d_{uv})^{-2}\) for \textit{SM}.

Algorithm 2 describes the details of \textit{ShSM} using the pseudo code.
Algorithm 1: ShFR

Input: Graph $G = (V, E)$, # of iterations $k$

repeat $k$ times
  // Initialize displacement
  for $u \in V$ do
    $x'_u, y'_u = 0$;
  end
  // FR repulsion force
  for $u \in V$ do
    for $v \in V, v \neq u$ do
      $x'_u + = \frac{x_v - x_u}{||x_v - x_u||^2}fl^2$;
      $y'_u + = \frac{y_v - y_u}{||x_v - x_u||^2}fl^2$;
    end
  end
  // FR attraction force
  for $e = (u, v) \in E$ do
    $x'_u = (x_u - x_v)||(x_v - x_u)||/l$;
    $y'_u = (y_u - y_v)||(x_v - x_u)||/l$;
    $x'_v = (x_v - x_u)||(x_v - x_u)||/l$;
    $y'_v = (y_v - y_u)||(x_v - x_u)||/l$;
  end
  // Update coordinates of vertices in $V$ according to the displacement
  for $u \in V$ do
    $x_u + = x'_u$;
    $y_u + = y'_u$;
  end
Compute proximity graph $S = (V, E')$;

// Initialize displacement
for $u \in V$ do
  $x_u, y_u = 0$;
end
// ShFR proximity forces
for $e = (u, v) \in E \setminus E'$ do
  $m$: mid-point of $u$ and $v$;
  // ShFR proximity repulsion force
  for $t \in V$ where $(u, t) \in E' \setminus E$ or $(v, t) \in E' \setminus E$ do
    $x'_t + = \frac{x_t - x_m}{||x_t - x_m||^2}fl^2||x_t - x_u||$;
    $y'_t + = \frac{y_t - y_m}{||x_t - x_m||^2}fl^2||x_t - x_u||$;
  end
  // ShFR proximity attraction force
  $x'_u - = (x_u - x_v)||(x_v - x_u)||/2l$;
  $y'_u - = (y_u - y_v)||(x_v - x_u)||/2l$;
  $x'_v = (x_u - x_v)||(x_v - x_u)||/2l$;
  $y'_v = (y_u - y_v)||(x_v - x_u)||/2l$;
end
// Update coordinates of vertices in $V$ according to the displacement
for $u \in V$ do
  $x_u + = x'_u$;
  $y_u + = y'_u$;
end
Algorithm 2: ShSM

Input: Graph $G = (V, E)$, # of iterations $k$

1. $d_{uv} \leftarrow \text{ShortestPaths}(G)$;
2. Compute weights $w_{uv}$;
3. Compute initial layout $D$ of $G$ using PivotMDS;
4. repeat $k$ times
5. // SM stress
6. for $u \in V$ do
7. // Initialize weight sum and displacement
8. $W_u = 0$;
9. $x_u', y_u' = 0$;
10. // Stress minimization computation
11. for $v \in V, v \neq u$ do
12. $x_u' + = w_{uv}(x_u) + d_{uv}(x_v - x_u)/||X_v - X_u||$;
13. $y_u' + = w_{uv}(y_u) + d_{uv}(y_v - y_u)/||X_v - X_u||$;
14. $W_u + = w_{uv}$;
15. end
16. // Update coordinates of vertices in $V$ according to the displacement
17. $x_u = x_u'/W_u$;
18. $y_u = y_u'/W_u$;
19. end
20. Compute proximity graph $S = (V, E')$;
21. // Initialize weight sum and displacement
22. for $u \in V$ do
23. $W_u = 0$;
24. $x_u', y_u' = 0$;
25. end
26. // ShSM proximity stress
27. for $e = (u, v) \in E \setminus E'$ do
28. $m$: mid-point of $u$ and $v$;
29. // ShSM proximity repulsion stress
30. for $t \in V$ where $(u, t) \in E \setminus E$ or $(v, t) \in E' \setminus E$ do
31. $x_t' + = w_{uv}(x_m) + d_{uv}(x_t - x_m)/||X_m - X_t||/||X_t - X_m||$;
32. $y_t' + = w_{uv}(y_m) + d_{uv}(y_t - y_m)/||X_m - X_t||/||X_t - X_m||$;
33. $W_t + = w_{uv}$;
34. end
35. // ShSM proximity attraction stress
36. $x_u' + = (w_{uv})^k(x_u) + d_{uv}(x_v - x_u)/||X_v - X_u||$;
37. $y_u' + = (w_{uv})^k(y_u) + d_{uv}(y_v - y_u)/||X_v - X_u||$;
38. $W_u + = (w_{uv})^k$;
39. $x_v' + = (w_{uv})^k(x_v) + d_{uv}(x_u - x_v)/||X_u - X_v||$;
40. $y_v' + = (w_{uv})^k(y_v) + d_{uv}(y_u - y_v)/||X_u - X_v||$;
41. $W_v + = (w_{uv})^k$;
42. end
43. // Update coordinates of vertices in $V$ according to the displacement
44. for $u \in V$ do
45. $x_u = x_u'/W_u$;
46. $y_u = y_u'/W_u$;
47. end
48. end
Appendix E  *ShFR* and *ShSM*: Visual Comparison

**Table 9.** Visual comparison of *FR* and *ShFR, SM* and *ShSM* on strong proximity drawable graph classes and synthetic scale-free graphs. *ShFR* manages to untangle strong proximity drawable graphs better than *FR*. Meanwhile, *ShSM* manages to “open up” the faces collapsed by *SM* and highlights dense areas in scale-free graphs better.

|        | *FR* | *ShFR* | *FR* | *ShFR* | *FR* | *ShFR* |
|--------|------|--------|------|--------|------|--------|
| Max. outerplanar | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Biconn. outerplanar | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Tree | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 2$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 3$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 5$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |

*SM* and *ShSM*: Visual comparison of *SM* and *ShSM* on strong proximity drawable graph classes and synthetic scale-free graphs. *ShSM* manages to “open up” the faces collapsed by *SM* and highlights dense areas in scale-free graphs better.

|        | *SM* | *ShSM* | *SM* | *ShSM* | *SM* | *ShSM* |
|--------|------|--------|------|--------|------|--------|
| Max. outerplanar | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Biconn. outerplanar | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Tree | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 2$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 3$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |
| Scale-free ($d = 5$) | ![image] | ![image] | ![image] | ![image] | ![image] | ![image] |