DOI: 10.1111/cgf.70127
Eurographics Conference on Visualization (EuroVis) 2025
W. Aigner, N. Andrienko, and B. Wang
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 44 (2025), Number 3

Viewpoint Optimization for 3D Graph Drawings

S. van Wageningen[1] and T. Mchedlidze[1] and A. Telea[1]

[1]Utrecht University, Department of Information and Computing Sciences, The Netherlands

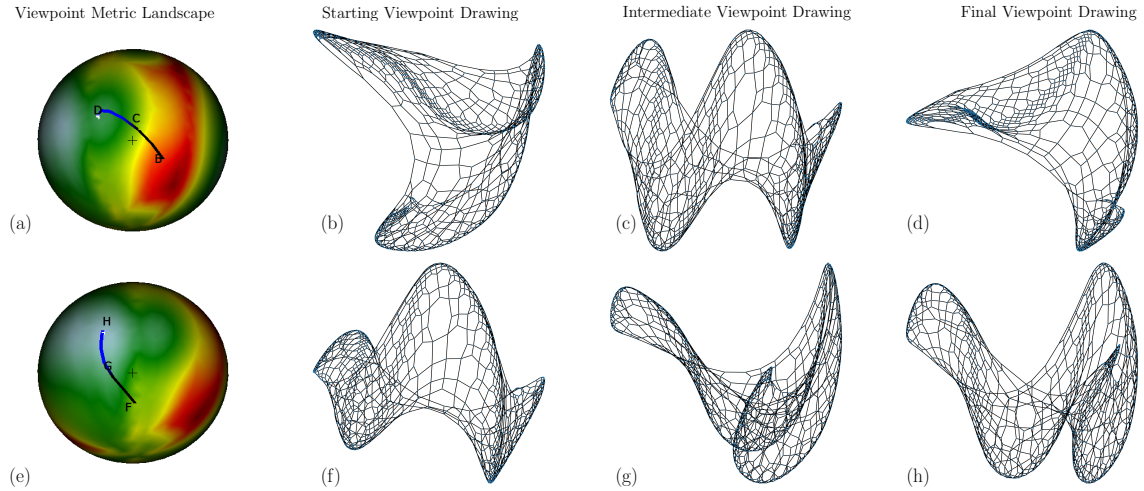

Figure 1: *Gradient descent traversing the metric landscape of viewpoints of a 3D graph drawing. (a) and (e) show metric landscapes of the same 3D drawing with different paths to an optimum of uniform edge lengths. (b), (c), (d), and (f), (g), (h), show the starting, intermediate, and final viewpoint drawings found by gradient descent of (a) and (e), respectively.*

Abstract

Graph drawings using a node-link metaphor and straight edges are widely used to represent and understand relational data. While such drawings are typically created in 2D, 3D representations have also gained popularity. When exploring 3D drawings, finding viewpoints that help understanding the graph's structure is crucial. Finding good viewpoints also allows using the 3D drawings to generate good 2D graph drawings. In this work, we tackle the problem of automatically finding high-quality viewpoints for 3D graph drawings. We propose and evaluate strategies based on sampling, gradient descent, and evolutionary-inspired meta-heuristics. Our results show that most strategies quickly converge to high-quality viewpoints within a few dozen function evaluations, with meta-heuristic approaches showing robust performance regardless of the quality metric.

CCS Concepts

• ***Human-centered computing*** → ***Graph drawings;*** • ***Computing methodologies*** → *Genetic algorithms; Neural networks;*

1. Introduction

Graph drawings, also known as node-link diagrams, are common visualizations used to understand relational data. As the quality of such a drawing goes up, so does user task performance and data understanding [PCJ96], so a key goal in this field is the creation of drawings of the highest quality. To measure the quality of such drawings, many metrics have been developed that try to capture various aesthetic preferences of users [PCJ96, Hua07].

Graph drawings are most commonly created and explored in 2D. Although 3D graph drawing is not new – one of the first graph drawing algorithms actually supported 3D outputs [FR91] – it is still far less popular than 2D graph drawing due to limited evidence for its advantages [WF96, WM08, FPK*23]. Unlike 2D drawings, 3D drawings require finding good viewpoints to explore the drawing from. In our previous work [vWMT24], we measured the quality of *viewpoint drawings* – that is, 2D views of a 3D graph drawing obtained from a given viewpoint – considering attributes like node

positions, edge lengths and angles, node/edge occlusion, and edge crossings. We have shown that viewpoint drawings can yield higher quality than when directly drawing the same graphs in 2D. This brings additional value to 3D graph drawings as tools for generating good 2D drawings (rather than being an end product by themselves): As 3D drawings can be computed as easily (and quickly) as 2D drawings, if we can *automatically* find 2D views of 3D drawings that have higher quality than using 2D layout algorithms, it means we can ultimately generate better 2D graph drawings.

Finding such high-quality viewpoints is a relatively unexplored problem. Only a few studies have attempted it resulting in heuristics with large time complexities [EHW97, HW98]. These heuristics have varying definitions of quality which influence what is a good (viewpoint) drawing. In our work [vWMT24], we used a brute-force sampling of the 3D viewpoint space to find high-quality viewpoints, which is slow and of limited accuracy. Viewpoint optimization is more prevalent in other fields, *e.g.*, in graphics, where it is also known as *camera optimization*. Many strategies [KK88, PB96, FCOL99, VFSH01] have been used to efficiently search for the best viewpoint of a graphics scene with respect to some predefined quality measure such as object occlusion. However, such algorithms do not directly apply to 3D graph drawing.

In this work, we aim to design algorithms that find such high-quality viewpoints for 3D straight-line graph drawings using a node-link metaphor – that is, graph drawings in which nodes are represented as 3D points and edges are straight-line segments between such points – *efficiently* and *accurately*. We measure viewpoint quality using several quality metrics commonly found in 2D graph drawing literature. Our contributions are as follows:

- We propose several optimization algorithms to find such viewpoints and compare their efficiency and accuracy.
- Our results show that we can find high-quality viewpoints fully automatically, for a wide family of graphs and metrics, within a few seconds.

Combined with the findings in [vWMT24], this means we can create 2D straight-line, node-link type graph drawings with quality that can exceed that given by 2D straight-line, node-link type layout algorithms fully automatically and with a low cost.

The structure of the paper is as follows. Section 2 details related work. Section 3 details our optimization methods. Sections 4 and 5 describe and discuss the results obtained by our methods. Finally, Section 6 concludes the paper.

2. Related Work

2.1. 3D graph drawing

Many algorithms can create straight-line graph drawings in 3D, starting with the early force-directed Fruchterman-Reingold algorithm [FR91]. Later on, algorithms using simulated annealing targeted both 2D and 3D drawing [MRS96, CT96, JVHB14]. More recently, dimensionality reduction was used to draw graphs in both 2D and 3D [GKN05, BP07, KRM*17].

A few studies have examined the (dis)advantages of 3D graph drawing. Ware and Franck [WF96] performed a study in which users had to find connections in a 3D graph drawing. Users who could navigate these 3D drawings had fewer error rates but longer decision times compared to using 2D drawings. For path length tasks, the users' error rate was much lower for 3D drawings when using motion and/or stereoscopic depth cues [WM08]. User studies performed in Virtual Reality show mixed results [FPK*23], with best user performances depending on the task type and number of dimensions (2D vs 2.5D vs 3D) used for display. Such mixed findings were also found in other visualization types, with 3D outperforming 2D [AWR18] and conversely [BGP*11].

Greffard et al. [GPK11] studied the task performance for community detection across 2D, 3D, and 3D with stereoscopic depth cues displays. Good results were found for stereo 3D for larger graphs; for smaller graphs, 2D showed a lower error rate. Interestingly, the average response time was always lowest for 2D drawings, regardless of graph complexity and size. These response-time findings mirror Eades et al. [EHW97] who conjectured that this trade-off occurs due to users trying to find favorable viewpoints. This further supports a need for automatic search of good viewpoints.

2.2. Viewpoint optimization

Graph drawing: Early works that looked into viewpoint optimization for graph drawing measured viewpoint quality via the amount of vertex and/or edge occlusions [EHW97, HW98]. Eades et al. [EHW97] proposed a Voronoi-based algorithm which moves a starting viewpoint away from near bad viewpoints to find the theoretical best viewpoint. Houle and Webber [HW98] improved on this by approximating, rather than finding the exact, best viewpoint. Such methods were created for specific measures of what is a good viewpoint, and cannot be easily extended to other quality metrics.

In our previous work [vWMT24], we measured the quality of a large sample of viewpoints of a 3D graph drawing by standard quality metrics for 2D graph drawings. We found that such viewpoints – or, more precisely, the 2D perspective projections of the 3D drawing taken from those viewpoints – can yield higher quality than drawing the same graphs using the same layout algorithms run in 2D. Castelein et al. [CTMT23] used a related approach to find good viewpoints of 3D scatterplots obtained via dimensionality reduction; and reported higher quality than when projecting the high-dimensional data directly in 2D. Both above works find high-quality viewpoints by uniformly and densely sampling the viewpoint space, evaluating quality metrics, and returning the highest value(s) obtained. This brute-force method is obviously slow as thousands of viewpoints must be used for good results. To our knowledge, no method exists that *efficiently* and *accurately* approximates the best viewpoint of a 3D graph drawing w.r.t. conventional quality metrics.

Computer graphics – quality metrics: Various metrics have been proposed to define optimal viewpoints for a 3D scene rendering. Kamada and Kawai [KK88] aimed to find the *general position*, that is, the viewpoint from which a 2D image of a 3D scene retains the most shape information present in the scene. Plemenos and Benayada [PB96] aimed to maximize the number of faces and the surface area of a 3D scene's projection to 2D. Fleishman et al. [FCOL99] studied scene visibility using reference views. Measures such as entropy [VFSH01] and the Kullback-Leibler divergence [SPFG05, MSC24] were also used to find the best viewpoint

for 3D scenes. Bonaventura et al. [BFS*18] neatly summarize the many measures used for viewpoint selection in computer graphics.

Computer graphics – optimization strategies: Several methods have been proposed to find high-quality viewpoints given by the metrics outlined above. In all cases, viewpoints lie on a sphere surrounding the viewed 3D scene, and look at the sphere's center, following the well-known 'world in hand' metaphor. Kamada and Kawai [KK88] proposed, among others, a uniform covering sampling approach which was also used by Vazquez et al. [VFSH01]. Plemenos and Benayada [PB96] proposed a refined sampling which splits the sphere into eight spherical triangles. The best triangle is then selected and the procedure is repeated recursively.

Finding good-quality viewpoints can also be seen as optimizing a quality function whose parameters model the viewpoint. To find the global optimum, methods such as simulated annealing [Stü98], particle swarm optimization (PSO) [WZZ*07, YZY18], and genetic algorithms [ZY22] have been used. Simulated annealing relies on a global temperature, initial viewpoint, and a cooling factor, to search for the optimum. PSO searches by using a set of particles that communicate with each other to share information about the solution space. A genetic algorithm has multiple individuals in its population, similar to PSO particles, but 'shares' information with other individuals only via mutation, selection, and crossover operations. While all these strategies have some advantages, *e.g.*, not requiring the target function to be differentiable, they all are metaheuristics which are not guaranteed to find the global optimum.

Machine learning techniques can also be used for viewpoint optimization. Gradient descent has been used to travel to the best viewpoint [LVJ05, WW22, HG24]. This method, however, can get stuck into local optima. To alleviate this, a random starting viewpoint [LVJ05] and/or multiple runs with various starting viewpoints can be used [WW22, HG24]. Vieira et al. [VBP*09] used support vector machines to classify areas to find more suitable viewpoints. Yang et al. trained a *Convolutional Neural Network* (CNN) on images of viewpoint samples of volume data. After training, the model can recommend good viewpoints. Zhang et al. [ZFY20] used a CNN with the inclusion of ground truth data. They fed the model an aesthetic reference image and a 3D scene and trained it to find viewpoints of a 3D scene that match the reference image.

3. Method

As discussed so far, many strategies have been used to approximate the highest-quality viewpoint in computer graphics. To our knowledge, none of these has been used for viewpoint optimization in graph drawing – a task which we address next.

We start by introducing a few notations and concepts following [vWMT24]. An *undirected graph* $G = (V,E)$ is a set of nodes $V = \{v_1, \ldots, v_n\}$ and a set of undirected unweighted edges $E = \{e_1, e_2, \ldots, e_m\} \subseteq V \times V$. A *graph drawing*, or *layout*, Γ of G assigns k-dimensional coordinates, $k \in \{2,3\}$, to nodes in V, and can be represented as a matrix $X^k \in \mathbb{R}^{n \times k}$ where row $X_i \in \mathbb{R}^{1 \times k}$ gives the coordinates of node v_i. Edges are drawn as straight-lines. Let $D \in \mathbb{R}^{n \times n}$ denote the *shortest-path matrix* of graph-theoretic distances d_{ij} between all node-pairs (v_i, v_j) in $V \times V$.

We next refer to the Euclidean distance $\|X_i - X_j\|$ between nodes v_i and v_j simply as *distance*. Let $deg(v)$ be the *degree* of node v, *i.e.*, the number of edges incident to v; let $L(e)$ denote the length of edge e, *i.e.*, distance of its endpoints in Γ. Finally, a *quality metric* is a function $Q(\Gamma) \in [0,1]$ that assigns a value to the drawing Γ of G, with low (resp. high) values denoting poorer (resp. better) drawings.

From any 3D drawing Γ of a graph G, we can generate 2D drawings as follows. Consider a sphere S enclosing Γ. A *viewpoint* $p = (\theta, \phi)$ is a location on S given by spherical coordinates, angles θ and ϕ. The *viewpoint drawing* $\Gamma(p)$ is the perspective projection of Γ with camera placed at p and looking at the sphere center.

3.1. Quality Metrics

To gauge the quality of a 2D graph drawing Γ, we use several quality metrics Q following their popularity in related work, their usefulness in assessing the quality of viewpoints of 3D drawings [EHW97], and user studies showing their ability to gauge quality as perceived by humans [PCJ96, Hua07].

Stress: This metric [KK89] has been shown to correlate with users' preferences [CEE*14] and measures how much all node-pair distances in a drawing deviate from their shortest path distances as

$$\texttt{ST}(\Gamma) = 1 - \frac{1}{n(n-1)/2} \sum_{i<j}^{n} \frac{(\|Z_i - Z_j\| - d_{ij})^2}{d_{ij}^2}, \tag{1}$$

where Z_i scales the coordinate X_i by the shortest path distances as

$$Z_i = \frac{\sum_{i \neq j} \|X_i - X_j\|/d_{ij}}{\sum_{i \neq j} \|X_i - X_j\|^2/d_{ij}^2} X_i. \tag{2}$$

Edge length deviation: Users favor consistent edge lengths in a drawing [CP96]. We thus compute an *Edge length deviation* to gauge the average deviation of edge lengths from the mean μ of all such lengths in a drawing as

$$\texttt{ELD}(\Gamma) = 1 - \sqrt{\frac{1}{m} \sum_{i=1}^{m} (L(e_i) - \mu)^2}. \tag{3}$$

Crossing number: Purchase et al. [PCJ96] show that the number of crossings affects how well a drawing can be understood. We measure the number of crossings C_{cnt} normalized by its maximum possible value C_{poss} by

$$\texttt{CN}(\Gamma) = 1 - \frac{C_{\text{cnt}}}{C_{\text{poss}}}, \tag{4}$$

where the maximum possible number of crossings C_{poss} is given by

$$C_{\text{poss}} = \frac{m(m-1)}{2} - \frac{1}{2} \sum_{i=1}^{n} (deg(v_i)(deg(v_i) - 1)). \tag{5}$$

As C_{cnt} is discrete, $\texttt{CN}(\Gamma)$ is also discrete. This discreteness is incompatible with gradient-based optimization strategies we will next use (Sec. 3.3). To solve this, we replace C_{cnt} by the differentiable proxy computed by the Neural Aesthete [TCG22, ALD*22], a *Multilayer Perceptron* (MLP) trained on edge crossing classification with a reported 97% accuracy. We use the exact architecture in [TCG22]. We optimize hyperparameters by grid search and find that a learning

rate of 0.05 and a momentum of 0.9 increase accuracy to 98.5%. The model is trained for 20 epochs with a batch size of 1024 on a randomly generated dataset of 1e6 edge coordinates.

Node-node occlusion: The original occlusion metrics in [EHW97] measure how often nodes and edges perfectly overlap. We slightly alter this metric by computing the closeness of pairs of nodes as

$$\texttt{NN}(\Gamma) = 1 - \frac{1}{n(n-1)/2} \sum_{i \neq j} \frac{\|X_i - X_j\| - d - \big|\|X_i - X_j\| - d\big|}{2d}, \tag{6}$$

where d is the diameter of a node set to a default of $\frac{1}{\sqrt{n}}$. When all nodes are stacked atop each other, $\texttt{NN}(\Gamma) = 0$. When all nodes are at a distance of at least d from every other node, $\texttt{NN}(\Gamma) = 1$.

Node-edge occlusion: We treat node-edge occlusion similarly to NN. We measure the closeness of all compatible node-edge pairs as

$$\texttt{NE}(\Gamma) = 1 - \frac{1}{nm - \sum_{i=1}^{n} deg(v_i)} \sum_{i}^{n} \sum_{j}^{m} \frac{\|X_i - Y_j\| - d - \big|\|X_i - Y_j\| - d\big|}{2d}, \tag{7}$$

where Y_j is the point on edge j to which node i is closest to.

Combinations: High-quality drawings can be produced by optimizing several quality metrics [HEHL13]. Quality metrics for graph drawings tend to correlate indicating that not all metrics need be jointly optimized [MPWK24]. To find suitable combinations we first perform a simple linear and non-linear correlation analysis of the metrics over the viewpoints and graphs in [vWMT24]. This analysis shows that only the Node-node occlusion and Node-edge occlusion strongly correlate (Figure 2). We therefore propose a linear combination LC of Stress (ST), Edge length deviation (ELD), Node-node occlusion (NN), and the Crossing number (CN). Figure 2 also indicates that the sole optimization of one metric, such as ST, may lead to worse values of other metrics, such as NN.

3.2. Viewpoint Landscape

We next use *viewpoint landscape spheres* to get an intuition of how quality metrics vary over viewpoints. Consider the sphere S used to construct viewpoint drawings (Sec. 3). For each point $p \in S$, we color p with the value of $Q(\Gamma(p))$, *i.e.* encode there the quality of the 2D viewpoint drawing $\Gamma(p)$ measured by a quality metric Q. For implementation details, we refer to [CTMT23, vWMT24]. Figure 3 shows the spheres for the five quality metrics in Sec. 3.1. We see that ST, ELD, NN, and NE have quite smooth variations from their minima to their maxima; in contrast, CN is less smooth, showing many local extrema. Hence, an optimization *strategy* should consider the type of metric. For instance, we conjecture that local search optimizations like gradient descent may work better for ST, ELD, and NE; while global search optimizations, *e.g.* random sampling and evolutionary inspired meta-heuristics, may perform better on CN. We explore this conjecture in Sec. 4.

3.3. Viewpoint Optimization

We next present the optimization strategies we designed to automatically find high-quality viewpoints, inspired by commonly used

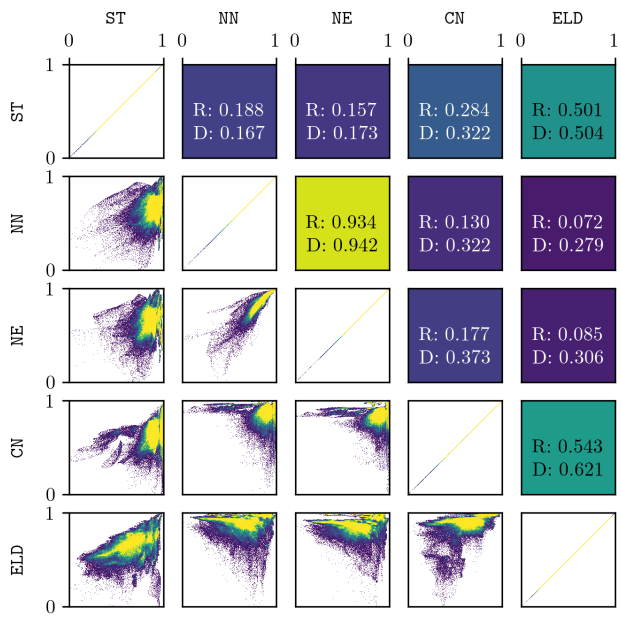

Figure 2: *Density plots of pairs of normalized quality metrics (lower triangular matrix; purple: low density; yellow: high density). Pearson's R correlation (linear) and distance correlation D (non-linear) of pairs of quality metrics (upper triangular matrix).*

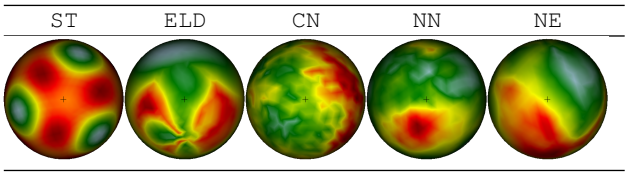

Figure 3: *Viewpoint landscapes of 5 metrics for 3D drawings created by ForceAtlas2. The graphs being drawn are, from left to right: Sierpinski3D, grafo10229, grafo10228, grafo10226 and netscience.*

strategies. Pseudocode and parameter values are given in the supplementary material – see next references marked with ⋆. Source code of all strategies, datasets, and analysis is available online.

The search space for all techniques only covers half of the viewpoint sphere given the symmetry of the 2D projection operation. We measure the *effectiveness* of our strategies by comparing their results with *ground-truth values* (discussed later in this section). We measure strategy *efficiency* by counting the number N of needed metric evaluations, where we cap $N \leq N_{\max}$, with $N_{\max} = 500$.

Uniform Sample: We use Fibonacci sampling to create N viewpoints $P = \{p_1, \ldots, p_N\}$ that uniformly cover the half-sphere. Other strategies can be used to create similar uniform results *e.g.* icosahedral subdivision often met in light field computations [OTWO17]. We measure their quality and keep the best one (Alg. 1⋆).

Iterative Resampling: We propose a new iterative sampling strategy inspired by existing heuristics [PB96, HW98]. We start like

UNIFORM SAMPLE by computing an initial set of N viewpoints P_0. From P_0, we take the best viewpoint p_{best} and set it as the center of a new region of interest. We then find the closest viewpoint $p_{\text{next}} \in P_0$ to p_{best} in terms of Euclidean distance. We create a rectangular sampling grid P_1 on the sphere centered at p_{best} and of half-diagonal size $\|p_{\text{next}} - p_{\text{best}}\|$ containing N samples. We repeat this sampling for L iterations, yielding the viewpoint-sets $PS = \{P_0, \ldots, P_L\}$ of size $(N + N * L) = N_{\max}$ (Alg. $2^\star$). UNIFORM SAMPLE and ITERATIVE RESAMPLING are jointly called *sampling strategies*.

Simulated Annealing-1/5: This strategy uses a global temperature and a cooling effect to stochastically find an optimum. Simulated annealing iteratively searches for viewpoints p' close to the current one p. It then accepts or rejects p' depending on the quality of both p and p', the global temperature, and a random probability. By including randomness, the strategy may accept viewpoints with a worse quality, which can be good in the long run as we escape local minima. We use a more modern approach named *dual annealing*, implemented in SciPy [VGO*20], which combines classical and fast simulated annealing. Five strategy-specific parameters exist; of these, the starting viewpoint is the most important for finding a local or the global optimum. Given the starting viewpoint influence, we test two versions: SIMULATED ANNEALING-1, where we use a random starting viewpoint, and SIMULATED ANNEALING-5, where we do simulated annealing from five starting viewpoints $p^5 : \{0,0\}, \{\frac{\pi}{4},0\}, \{0,\frac{\pi}{4}\}, \{0,-\frac{\pi}{4}\}$, and $\{-\frac{\pi}{4},0\}$.

Differential Evolution: Similar to genetic algorithms, DIFFERENTIAL EVOLUTION uses several (random) starting viewpoints as a population. Via crossover and a differential weight (mutation) it creates new candidate solutions that replace solutions in the starting population if they have the same or higher quality. During crossover, a viewpoint's angles can be replaced by a linear combination of the corresponding angles of three other distinct random viewpoints. This strategy has six parameters of which the size of the population, the crossover rate, and the differential weight are most important. We use the SciPy [VGO*20] implementation.

Particle Swarm Optimization: Individual solutions/particles are used to search for an optimum. Each particle's velocity is computed by using, with a random probability, the particle's best solution found so far and the best solution found by the entire swarm. With this velocity, the particle then moves to a new solution. Depending on the solution quality, the particle's best solution and the swarm's best solution is then updated. This strategy has four parameters of which the size of the swarm is most important. We use the PySwarms [Mir18] library to implement this strategy. We classify SIMULATED ANNEALING, DIFFERENTIAL EVOLUTION, and PARTICLE SWARM OPTIMIZATION as *meta-heuristics*.

Gradient Descent-1/5 For differentiable objective functions, *gradient descent* computes the derivative and 'walks' from a starting position to an optimum. The *learning rate* parameter controls step sizes during this walk. This strategy has 3 parameters; of these, the starting viewpoint crucially influences whether the descent converges into a global or local optimum. We use PyTorch [PGM*19] to implement this strategy with the Adam optimizer (Alg. $3^\star$). Similar to SIMULATED ANNEALING-1/5, we use two variations of this strategy where we either take a random starting viewpoint (1) or use p^5 five fixed starting viewpoints (5).

Uniform Gradient Descent-V1/V2: This strategy combines UNIFORM SAMPLE and GRADIENT DESCENT and comes in two versions. Version 1 (`V1`) samples $n_p = 10$ starting viewpoints from the half sphere using Fibonacci sampling. GRADIENT DESCENT is then used from each starting viewpoint for $N_{\max}/n_p$ iterations. Version 2 (V2) evaluates the n_p starting viewpoints, selects the best one, and uses GRADIENT DESCENT for $N_{\max} - n_p$ iterations (Alg. $4^\star$).

Newton-Raphson-1/5: Like gradient descent, we use a derivative to go from a starting viewpoint to an optimum. However, we now use second-order derivatives instead of first-order ones. This can reduce the number of steps needed to reach an optimum. Yet, as known from numerical integration, this method can get stuck in so called 'saddle points'. This strategy has 5 parameters of which the most important is the starting viewpoint – see the SciPy [VGO*20] implementation. Similar to SIMULATED ANNEALING-1/5, we use two variations of this strategy where we either take a random starting viewpoint (1) or use p^5 five fixed starting viewpoints (5).

DeepGD: This *Graph Neural Network* (GNN) was originally designed to create high-quality 2D drawings [WYHS21]. We reuse the exact same architecture of DeepGD with a few modifications: We modify DeepGD to predict the ϕ and θ angles defining the highest-quality viewpoint for an unseen graph. For this, we change the input layer's dimensionality to accept 3D node coordinates. We add a pooling layer at the model's end to pool a predicted $nx2$ matrix of node coordinates into a 1×2 viewpoint vector. We also remove the ReLu activation from each layer in all blocks so that viewpoint prediction angles can be negative. Last, we design loss functions based on a quality metric Q and the perspective projection computation, so that the loss models how the predicted viewpoint influences Q. We train the model on the unstructured Rome graphs [GDT] for 25 epochs. We classify (UNIFORM) GRADIENT DESCENT, NEWTON-RAPHSON, and DEEPGD as *gradient strategies*.

Ground Truth: We compare all above strategies with the 'ground truth' acquired by running a very dense Iterative Resampling ($N_{\max} = 10000$, $L = 1$). Note that, even with this very dense sampling, we cannot guarantee that GROUND TRUTH captures the exact optimum of the studied quality metrics.

3.4. Datasets

We use our strategies on a set $\mathcal{G}$ of 45 graphs. In detail, we handpick a set of graphs from the *SuiteSparse Matrix Collection* [DH11], as these graphs tend to have recognizable 3D structures. We also use the graphs in Kruiger et al. [KRM*17] which strongly vary in size, density, and structure. Lastly, we use the *Rome* [GDT] graphs, a widely used as benchmark for 2D graph drawing [WYHS21, GLA*21]. $\mathcal{G}$ is described in detail in Sec. $1^\star$. The model used to approximate the crossing number `CN` (see Sec. 3.1) is quite slow for larger graphs. As such, we measure `CN` and the Linear combination `LC` on a subset $\mathcal{G}_s \subseteq \mathcal{G}$ of 19 graphs.

We create the 3D drawings for the graphs in $\mathcal{G}$ by adapting the

ForceAtlas2 (FA2) [JVHB14] layout technique, implemented in Python [Chi19], to output 3D coordinates. We run FA2 with the parameter values gravity 1.0, scaling ratio 2.0, jitter tolerance 1.0, and a maximum of 300 iterations, allowing us to produce layouts within seconds on a commodity PC machine.

4. Results

We next show the results of the 14 optimization strategies from Sec. 3.3 for the quality metrics in Sec. 3.1. We run each strategy for $N_{\max} = 500$ evaluations of the functions defining our quality metrics and record the obtained values at $N \in \{N_{10}, N_{20}, \ldots, N_{\max}\}$. As most strategies use probabilistic elements, we average their results over five runs for each graph in $\mathcal{G}$ or $\mathcal{G}_s$. The exceptions are UNIFORM SAMPLE and ITERATIVE RESAMPLING which are deterministic; we run these once for each N. The GROUND TRUTH is run once, whereas DEEPGD requires many function evaluations only during training. Actual running times for each strategy are given in Sec. 2*.

We also study *statistically significant differences* between the strategies. In this case, we compare the distribution of quality metric values of viewpoints found by strategy A and B after 40 or 300 function evaluations. Since our data is non-normally distributed and paired, we use the Wilcoxon Signed Rank test to test whether the distribution of strategy A is significantly greater than the distribution of strategy B. More details on all pairs of strategies and the exact p-values are given in Sec. 4* of the supplementary material.

Stress: Figure 4a shows Stress values over the number of function evaluations (N) averaged over all graphs in $\mathcal{G}$. We see that DEEPGD performs worst, with a difference of 0.04 to GROUND TRUTH. GRADIENT DESCENT-1 also performs poorly for $N \leq 200$, but improves to outperform NEWTON-RAPHSON-1 around $N = 300$. We next exclude GRADIENT DESCENT-1 and DEEPGD to focus on differences between better performing strategies (Figure 4b). We see that, for $N \leq 100$, NEWTON-RAPHSON-5 has the steepest increase. We also see that DIFFERENTIAL EVOLUTION converges towards the same values as NEWTON-RAPHSON-5.

To ensure that averaging over several graphs does not hide interesting facts, we next look at the distribution of ST values for all strategies, all graphs in $\mathcal{G}$, at a specific iteration N. Figures 5a and 5b show these distributions for $N = 40$ and $N = 300$ respectively. Matching previous findings, NEWTON-RAPHSON-5 produces on average, already for $N \geq 40$, much better viewpoints than all other strategies. After $N = 300$, the differences of ST between strategies become minimal. We also note some outliers – graphs with very low ST values. For NEWTON-RAPHSON-5, these are *grid* and *grafo10235*. These graphs have noticeably different structures; the former has a grid-like structure; the latter is a quite unstructured graph. Other graphs with similar structures did not pose a challenge for NEWTON-RAPHSON-5, so we can not hypothesize that the difference in performance is due to graph structure.

While metric-value differences between strategies are small, we stress that each metric has a different *effective distribution* in terms of values it reaches over the theoretical range $[0, 1]$. Hence, small metric-value differences could mean large differences in *drawings*. Figure 6a illustrates this. Some strategies such as UNIFORM SAMPLE, GRADIENT DESCENT-5, and NEWTON-RAPHSON-1/5 get

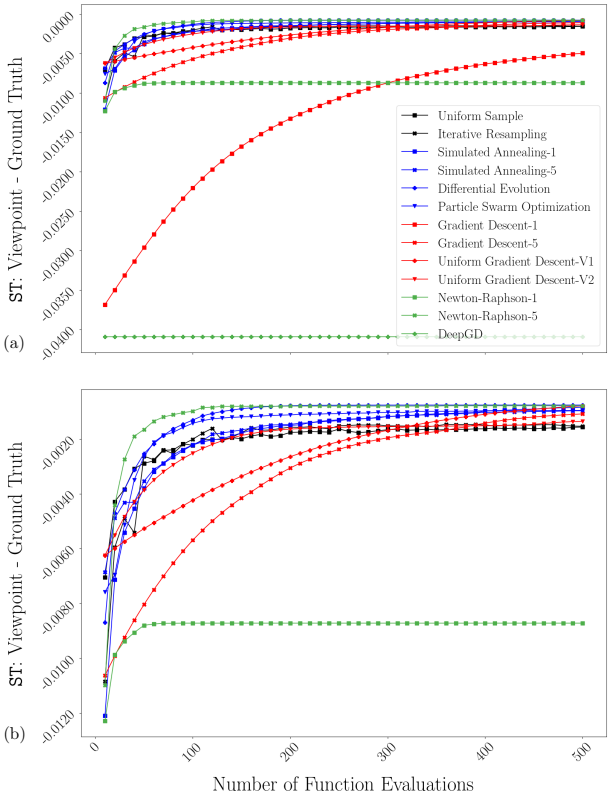

Figure 4: *Strategies performance* vs *Ground Truth over the number of function evaluations for Stress (ST) for all strategies (a), and all strategies excluding DeepGD and Gradient Descent-1 (b).*

quite close to the best viewpoint according to GROUND TRUTH. However, Figure 6b shows that the differences between viewpoints found at $N = 300$ are small, with most strategies (except DEEPGD) yielding the same output. We do find subtle differences between the clusters in the *sierpinski3d* graph *vs* crossings and occlusion at $N = 40$. The viewpoints found by different strategies are much more striking in the (lack of) symmetry for the *dwt_1005* graph.

Edge length deviation: For the remaining metrics we exclude GRADIENT DESCENT-1 and DEEPGD from the analysis due to their consistently poor results seen so far when compared to other strategies. For completeness, Sec. 3* shows figures including GRADIENT DESCENT-1, the distributions of viewpoints at $N = 40$ and $N = 300$, and the viewpoint drawings.

The first observation we make in Figure 7a is that for $N \geq 250$ NEWTON-RAPHSON-5 *exceeds* GROUND TRUTH. Also, similarly to the ST results, NEWTON-RAPHSON-5 converges much faster in the beginning compared to other strategies. We also see that the second best strategy is DIFFERENTIAL EVOLUTION. Similar to ST, Figures 18a* and 18b* show that these minor differences in ELD can lead to noticeable visual differences in the drawings.

Crossing number: As explained in Sec. 3.1, the crossing number CN is a discrete metric which we replace by a continuous approx-

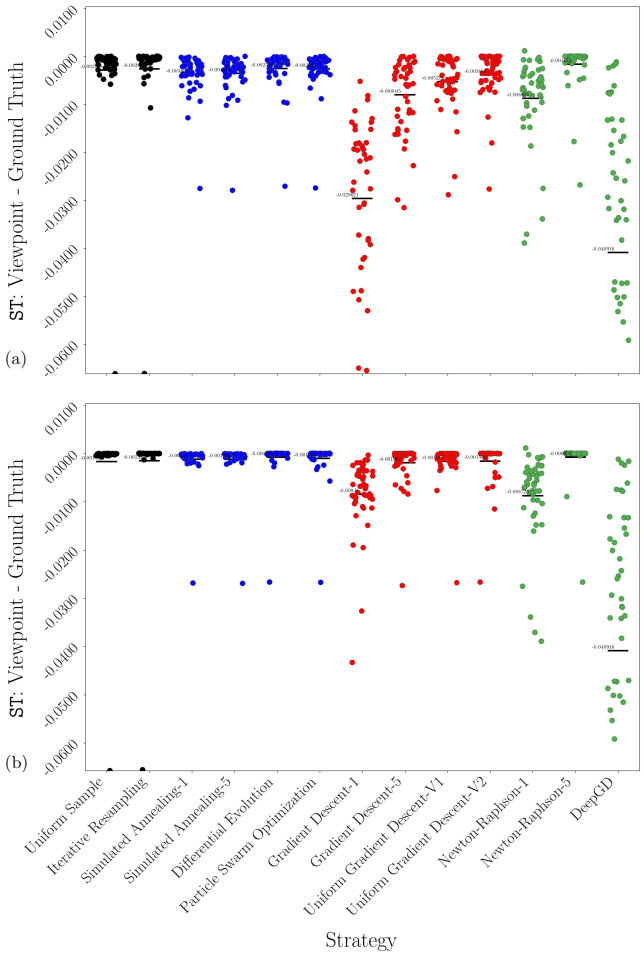

Figure 5: *Distribution of best viewpoints (w.r.t. Stress) found by each strategy after (a) $N = 40$ function evaluations and (b) $N = 300$ function evaluations. Black bars show mean values.*

imation to enable the use of gradient strategies. Figure 7b shows that, unlike its performance for the previous metrics, NEWTON-RAPHSON-5 performs poorly compared to other strategies. In fact, (meta-) heuristic and sampling strategies outperform gradient strategies, as the latter stagnate for $N \geq 120$. We also see that UNIFORM GRADIENT DESCENT-V1/V2 perform well for $N \leq 60$, after which they are quickly overtaken by DIFFERENTIAL EVOLUTION and PARTICLE SWARM OPTIMIZATION.

Figures 19a* and 19b* show that the viewpoints found for CN vary much more compared to viewpoints found for other metrics. We believe that this is due to the discrete nature of CN and the inaccuracy of the approximation used for CN in gradient strategies.

Node-node occlusion: Figure 7c shows the results for Node-node occlusion (NN). These are different from the previously discussed metrics. We see that the two sampling strategies UNIFORM SAMPLE and ITERATIVE RESAMPLING perform better than other strategies. UNIFORM SAMPLE leads until $N = 50$ and is then overtaken by ITERATIVE RESAMPLING which contests with DIFFERENTIAL EVOLUTION. Gradient strategies start off slow with only NEWTON-RAPHSON-5 performing well for $N \leq 90$.

The drawings shown in Figures 20a* and 20b* differ noticeably from the viewpoint drawings of other metrics. For example, we see that the optimization of NN leads to different viewpoint drawings as compared to those of ST in Figure 6b. Additionally, the strategies converge to different (local) minima leading to large visual differences, see *e.g.* graphs *GD96_c* and *can_96*.

Node-edge occlusion: Even though the NN and NE metrics are highly correlated, we find that strategies converge differently for Node-edge occlusion (NE). Figure 7d shows that DIFFERENTIAL EVOLUTION outperforms all other strategies for $N \geq 100$, and even outperforms GROUND TRUTH after roughly $N = 350$. However, the performance of DIFFERENTIAL EVOLUTION and the gradient strategies for $N \leq 100$ is comparable to the results for NN.

The actual viewpoint drawings for NE and NN (see Figs. 21a* and 21b*) show similar characteristics, which can be explained by the high correlation between these metrics. We again see more visual differences in the viewpoints found by our strategies for *GD96_c* and *can_96*, and with a stronger effect for $N = 40$ than for $N = 300$.

Linear combination: Figure 8 shows the strategy performance for LC. We see that meta-heuristic strategies outperform other strategies and converge to better results. Also, gradient strategies are outclassed by both meta-heuristic and sampling ones. We believe that these results are caused by the discrete CN metric, for which meta-heuristics work better than other strategies.

Figures 22a* and 22b* show the viewpoint drawings for Linear combination. For most graphs, strategies converge to similar viewpoints more than when only optimizing CN (see Figs. 19a*- 19b*).

Best strategies for all metrics: Since aggregating metrics into a linear combination may prioritize optimization of certain metrics due to a wider distribution of values, we also compare performance of strategies over all metrics. This gives us an alternate way to select a strategy that performs fairly well for multiple metrics. Figure 9 shows the performance of all strategies, except for DEEPGD and GRADIENT DESCENT-1, across the five quality metrics. We see that the results are in general consistent with the linear combination, thus meta-heuristics in general and DIFFERENTIAL EVOLUTION in particular tend to perform best.

5. Discussion

Exploration tactics: Our results show that strategies with more exploration tactics tend to find better results. Strategies that start from a single viewpoint have different issues: GRADIENT DESCENT-1 advances too slowly even with high learning rates; NEWTON-RAPHSON-1 advances faster but gets stuck in local minima. The exception here is SIMULATED ANNEALING-1 which uses a local-search algorithm to increase its search space. All other strategies explore from multiple starting viewpoints or use some form of information sharing, such as particles or population.

Discrete *vs* continuous metrics: Gradient strategies perform well

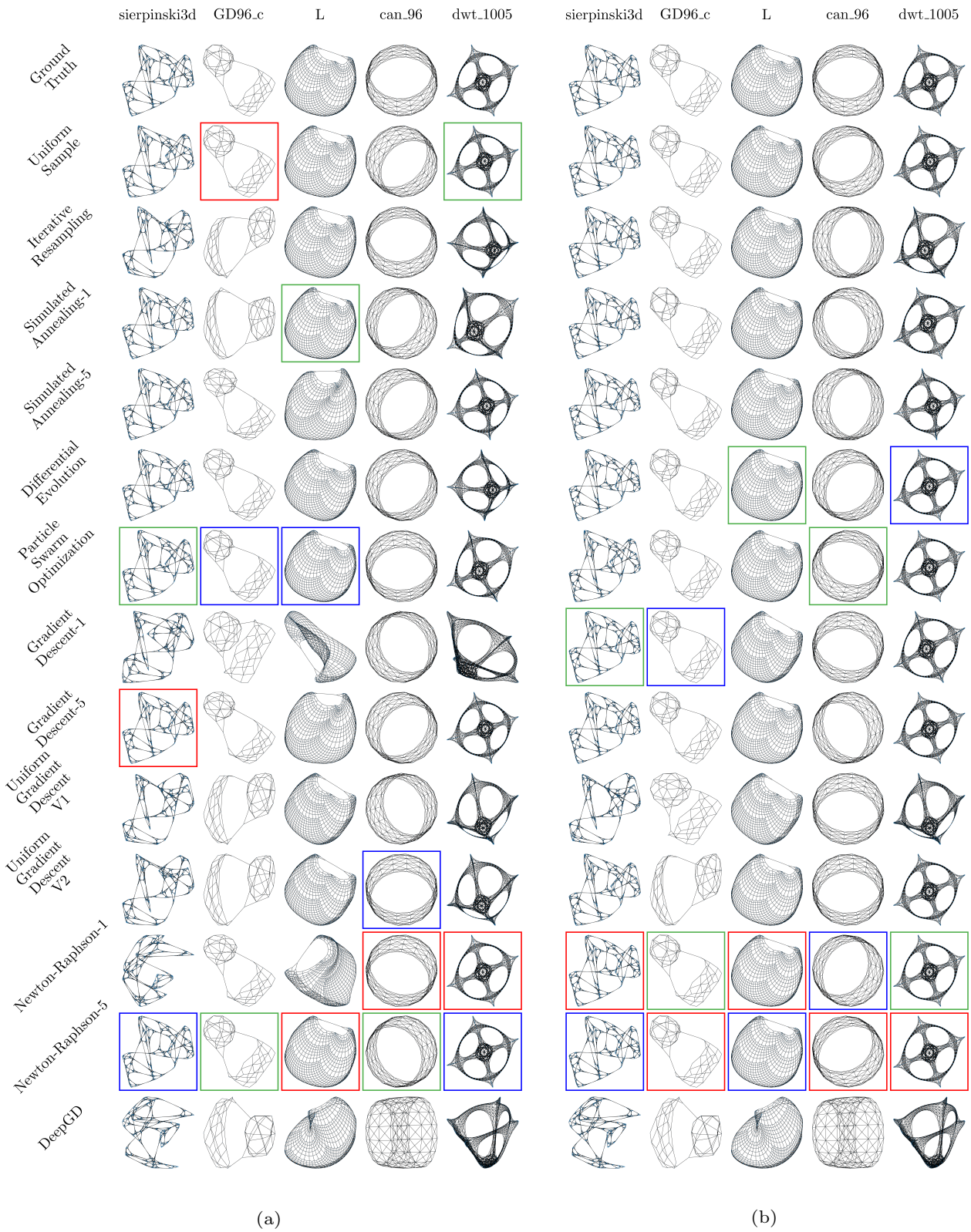

Figure 6: *Drawings of the best viewpoints (w.r.t Stress) found by all strategies after (a) 40 and (b) 300 function evaluations. The drawings outlined in red, blue, and green are metric-wise number 1, 2, and 3, respectively.*

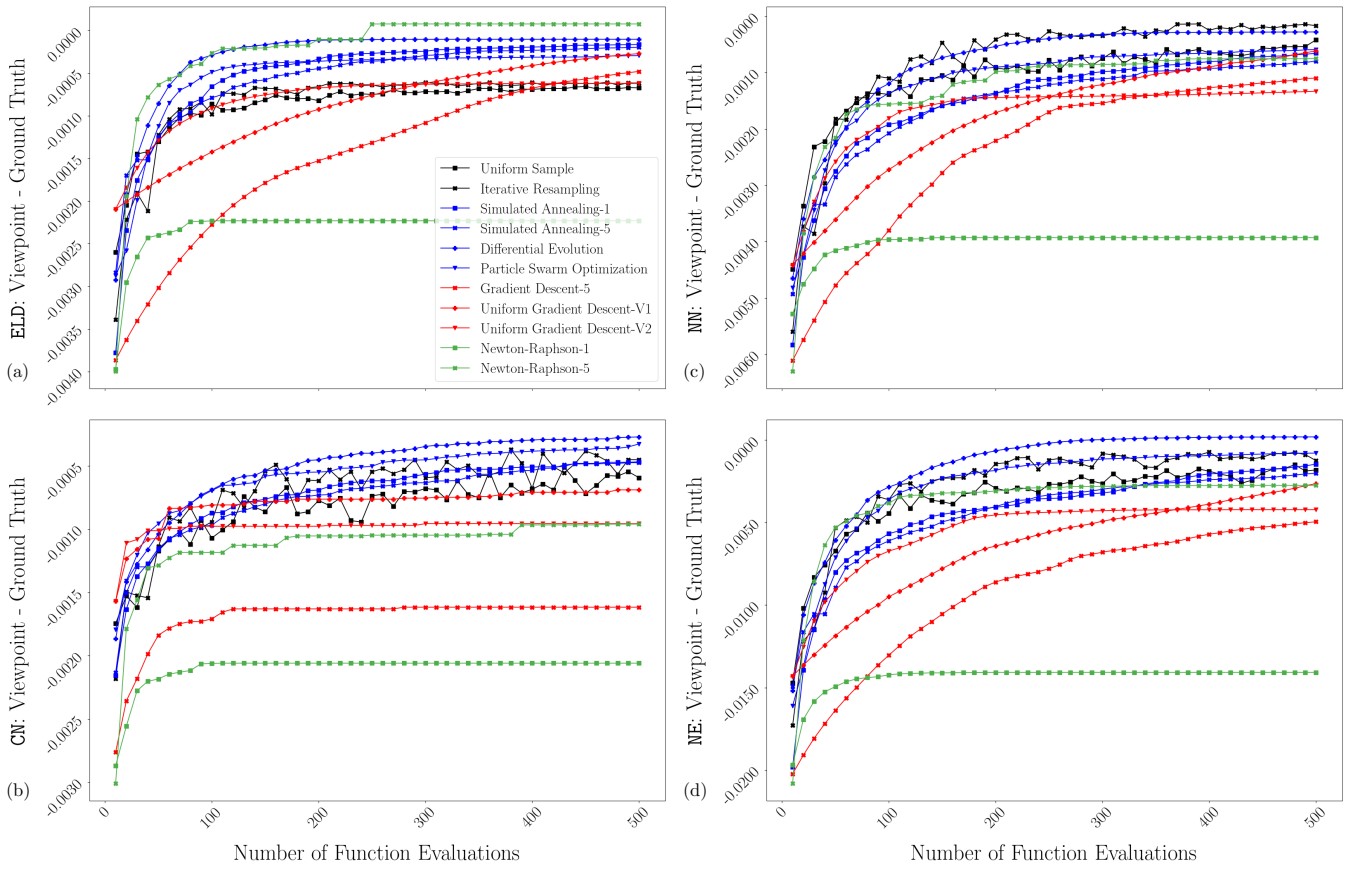

Figure 7: *Strategies performance* vs *Ground Truth over the number of function evaluations for Edge length deviation (ELD) in (a), Crossing number (CN) in (b), Node-node occlusion (NN) in (c), Node-edge occlusion (NE) in (d), excluding Gradient Descent-1.*

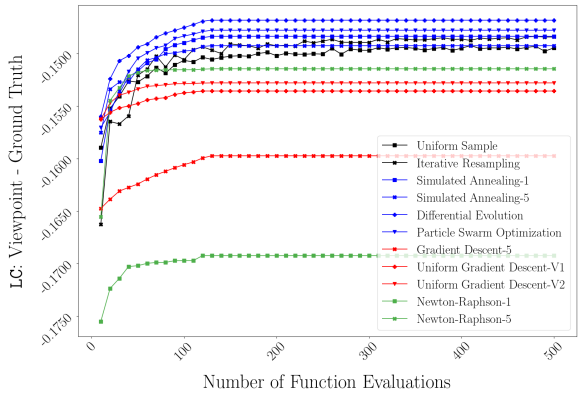

Figure 8: *Strategies performance* vs *Ground Truth over the number of function evaluations for a Linear combination (LC), excluding Gradient Descent-1.*

on continuous metrics but poorly on the discrete crossing number CN. *Meta-heuristic* strategies perform equally well for discrete and continuous metrics. We suspect that the effectiveness of gradient strategies for CN (and thus also for LC) is strongly hindered by the approximation we use for CN. Indeed, our MLP approximator has an accuracy of only $\sim 98\%$ for predicting if two edges cross. Even though this is a high value, this allows predicting viewpoints with fewer than optimal number of crossings which can be quite different than the optimal ones.

Practical value: Let us consider the efficiency of different strategies. NEWTON-RAPHSON-5 performs better than other strategies – it finds the best viewpoint for all continuous metrics within $N = 40$ evaluations. DIFFERENTIAL EVOLUTION consistently finds the best viewpoints for all metrics as N increases. Additionally, with the runtime information in Table 2* we can give guidance on what strategy is best in different scenarios. We see that either DIFFERENTIAL EVOLUTION or NEWTON-RAPHSON-5 is most ideal, when the goal is to find a high-quality viewpoint for any continuous metric as quickly as possible with the fewest number of evaluations. When the metic to be optimized is discrete, such as CN, the simple strategy ITERATIVE RESAMPLING is advised when speed is of importance. Last, when the goal is to acquire the best viewpoint possible, regardless of the metric and time cost, we see that DIFFERENTIAL EVOLUTION performs best.

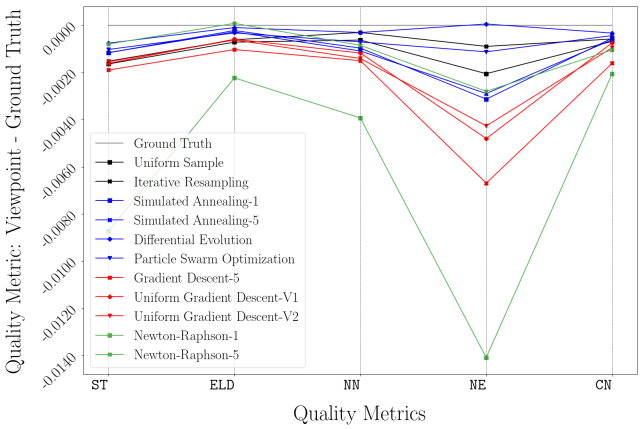

Figure 9: *Parallel coordinate plot of strategies performance* vs *Ground Truth for all metrics for* $N = 300$ *evaluations, excluding* LC *and Gradient Descent-1, averaged over all graphs and runs.*

The viewpoint drawings at $N = 40$ and $N = 300$ show that the graph and the optimized metric heavily influence how much the drawing changes as N increases. For 'simpler' graphs, such as the Rome ones, a good viewpoint can be found with as low as $N = 40$ evaluations. However, for more complex graphs, significantly better viewpoints can be found for higher N. Even though the change the in metric value may be small, the visual difference can be substantial, especially for graphs with salient 3D structures. From a *practical* perspective, we see that NEWTON-RAPHSON-5 and DIFFERENTIAL EVOLUTION find a high-quality viewpoint within *seconds* (on a commodity computer) even for graphs with more than 1000 nodes and edges. To get comparable results, sampling strategies may need many more function evaluations which leads to a *large* increase of the runtime, depending on the metric and graph. To conclude, our optimization proposal shows that one can fully automatically obtain high-quality 2D drawings of a wide range of graphs within seconds (atop the cost of running a 3D layout algorithm).

Limitations Our work is limited by a handful of factors. First, our strategy comparison is hindered by fixed parameter settings, as certain strategies such as GRADIENT DESCENT-1 involve randomness and may perform better or worse than others depending on their starting point. Also, parameters such as learning rate, temperature, population size, and mutation rate can alter the effectiveness of any strategy. Hence, our results of which strategy is best should not be taken as a definitive statement.

Second, the MLP used to approximate the crossing number limits the quality of the viewpoints predicted for this metric by gradient strategies. The metric space for the approximation of the crossing number is distinctly different as a result of the MLP inaccurately 'thinking' that a crossing is or isn't present when it should or shouldn't be. Furthermore, the runtime of the MLP becomes problematic as the size of the graph scales up. Lastly, our analysis is limited to only metrics meant for *straight-line* drawings. While our optimization strategies could perform well for drawings with curved edges, this is strongly influenced by the exact quality metrics that one would like to consider for such drawings. As such, and given the very large design space of curved-edge node-link drawings (and their quality metrics), we leave the study of viewpoint optimization for 3D curved-edge graph drawings to future work.

Finally, we only consider a subset of all existing quality metrics for (2D and 3D) graph drawing, chosen as Sec. 3.1 explains. Our work did not aim to focus on a rich analysis of all strategies *vs* all possible metrics. Rather, our goal was to check which strategies perform best on a few (but widely used) graph quality metrics. These best strategies can be next tested against additional quality metrics, *e.g.* symmetry, cliques, or advanced stress [WWS*17], in future work.

Future work Several directions for future work exist. First, the strategies used to search for best viewpoints could be improved. We still see potential in using a trained deep learning model to predict high-quality viewpoints for any 3D graph drawing. Its extremely fast inference is of great practical added value. To do this, we envisage further adapting the DEEPGD model as the current architecture seems excessive for a 'simple' problem as this. Moreover, building a new large dataset with a collection of graphs of different sizes, architectures, and characteristics is key to using such a model to predict viewpoints accurately for a wide variety of graphs. Additionally, some strategies could be combined to increase convergence speed, e.g. replacing gradient descent in UNIFORM GRADIENT DESCENT-V1/V2 with NEWTON-RAPHSON. Second, more research can be done towards finding more efficient approximation methods. The current MLP architecture is limited due to its accuracy and time complexity. Lastly, the perception of small metric changes is not well understood. Viewpoints found after 40 and 300 function evaluations can be distinctly different, but is the metric difference of these viewpoints important to a user? Do users perceive a viewpoint drawing to be truly better than another when their metrics differ by a small value? Answering this question is a topic for a user study.

6. Conclusion

We have presented an extensive evaluation of optimization strategies to find the best viewpoint of 3D graph drawings from the perspective or various well-known quality metrics. To our knowledge, this is the first attempt to find such viewpoints by optimization in the field of graph drawing. Our experiments show that certain strategies perform significantly better than others depending on the metric. Gradient strategies have subpar results for discrete metrics due to being limited by the MLP approximation technique. Non-gradient strategies perform well regardless of the metric nature. Out of all strategies, NEWTON-RAPHSON converges quickly but often gets stuck in local minima. The Differential Evolution strategy performs best overall. Our work shows that we can compute high-quality viewpoint drawings in less than 50 metric evaluations or, practically, a few seconds on a commodity PC. Increasing the number of metric evaluations can lead to improved drawings metric-wise. However, future work in terms of a user study is needed to determine whether these improved drawings correlate with increased human preference.

References

[ALD*22] AHMED R., LUCA F. D., DEVKOTA S., KOBOUROV S., LI M.: Multicriteria scalable graph drawing via stochastic gradient descent,

$(SGD)^2$. *IEEE Transactions on Visualization and Computer Graphics 28*, 06 (jun 2022), 2388–2399. doi:10.1109/TVCG.2022.3155564. 3

[AWR18] AYGAR E., WARE C., ROGERS D.: The contribution of stereoscopic and motion depth cues to the perception of structures in 3D point clouds. *ACM Transaction on Applied Perception 15*, 2 (feb 2018). doi:10.1145/3147914. 2

[BFS*18] BONAVENTURA X., FEIXAS M., SBERT M., CHUANG L., WALLRAVEN C.: A survey of viewpoint selection methods for polygonal models. *Entropy 20*, 5 (2018), 370. doi:10.3390/e20050370. 3

[BGP*11] BORKIN M., GAJOS K., PETERS A., MITSOURAS D., MELCHIONNA S., RYBICKI F., FELDMAN C., PFISTER H.: Evaluation of artery visualizations for heart disease diagnosis. *IEEE transactions on visualization and computer graphics 17* (12 2011), 2479–88. doi:10.1109/TVCG.2011.192. 2

[BP07] BRANDES U., PICH C.: Eigensolver methods for progressive multidimensional scaling of large data. In *Graph Drawing* (2007), vol. 4372 of *LNCS*, Springer Berlin Heidelberg, p. 42–53. doi:10.1007/978-3-540-70904-6_6. 2

[CEE*14] CHIMANI M., EADES P., EADES P., HONG S., HUANG W., KLEIN K., MARNER M., SMITH R. T., THOMAS B. H.: People prefer less stress and fewer crossings. In *Graph Drawing* (2014), Springer Berlin Heidelberg, pp. 523–524. 3

[Chi19] CHIPPADA B.: ForceAtlas2 Python, 2019. URL: https://github.com/bhargavchippada/forceatlas2. 6

[CP96] COLEMAN M. K., PARKER D. S.: Aesthetics-based graph layout for human consumption. *Software Practice and Experience 26*, 12 (dec 1996), 1415–1438. doi:10.1002/(SICI)1097-024X(199612). 3

[CT96] CRUZ I. F., TWAROG J. P.: 3D graph drawing with simulated annealing. In *Graph Drawing* (Berlin, Heidelberg, 1996), Brandenburg F. J., (Ed.), vol. 1027 of *LNCS*, Springer Berlin Heidelberg, p. 162–165. doi:10.1007/BFb0021800. 2

[CTMT23] CASTELEIN W., TIAN Z., MCHEDLIDZE T., TELEA A.: Viewpoint-based quality for analyzing and exploring 3D multidimensional projections. In *VISIGRAPP 2023* (Lisbon, Portugal, 2023), SCITEPRESS - Science and Technology Publications, p. 65–76. doi:10.5220/0011652800003417. 2, 4

[DH11] DAVIS T. A., HU Y.: The university of Florida sparse matrix collection. *ACM Transaction on Mathematical Software 38*, 1 (dec 2011). URL: https://sparse.tamu.edu/, doi:10.1145/2049662.2049663. 5

[EHW97] EADES P., HOULE M. E., WEBBER R.: Finding the best viewpoints for three-dimensional graph drawings. In *Graph Drawing* (Berlin, Heidelberg, 1997), vol. 1353 of *LNCS*, Springer Berlin Heidelberg. doi:10.1007/3-540-63938-1_53. 2, 3, 4

[FCOL99] FLEISHMAN S., COHEN-OR D., LISCHINSKI D.: Automatic camera placement for image-based modeling. In *Pacific Conference on Computer Graphics and Applications* (1999), p. 12–20. doi:10.1109/PCCGA.1999.803344. 2

[FPK*23] FEYER S. P., PINAUD B., KOBOUROV S., BRICH N., KRONE M., KERREN A., BEHRISCH M., SCHREIBER F., KLEIN K.: 2D, 2.5D, or 3D? An exploratory study on multilayer network visualisations in virtual reality. *IEEE Transactions on Visualization and Computer Graphics* (2023), 1–11. doi:10.1109/TVCG.2023.3327402. 1, 2

[FR91] FRUCHTERMAN T. M. J., REINGOLD E. M.: Graph drawing by force-directed placement. *Software: Practice and Experience 21*, 11 (1991), 1129–1164. doi:10.1002/spe.4380211102. 1, 2

[GDT] GDTOOLKIT: Rome graphs. URL: http://www.graphdrawing.org/data.html. 5

[GKN05] GANSNER E. R., KOREN Y., NORTH S.: Graph drawing by stress majorization. In *Graph Drawing* (Berlin, Heidelberg, 2005), vol. 3383 of *LNCS*, Springer Berlin Heidelberg, p. 239–250. doi:10.1007/978-3-540-31843-9_25. 2

[GLA*21] GIOVANNANGELI L., LALANNE F., AUBER D., GIOT R., BOURQUI R.: Deep neural network for drawing networks, (DNN^2). In *Graph Drawing* (Cham, 2021), Purchase H. C., Rutter I., (Eds.), Springer International Publishing, pp. 375–390. doi:10.1007/978-3-030-92931-2_27. 5

[GPK11] GREFFARD N., PICAROUGNE F., KUNTZ P.: Visual community detection: An evaluation of 2D, 3D perspective and 3D stereoscopic displays. In *Graph Drawing* (Berlin, Heidelberg, 2011), vol. 7034 of *LNCS*, Springer Berlin Heidelberg, p. 215–225. doi:10.1007/978-3-642-25878-7_21. 2

[HEHL13] HUANG W., EADES P., HONG S.-H., LIN C.-C.: Improving multiple aesthetics produces better graph drawings. *Journal of Visual Languages and Computing 24*, 4 (2013), 262–272. doi:10.1016/j.jvlc.2011.12.002. 4

[HG24] HIMMLER P., GÜNTHER T.: Transmittance-based extinction and viewpoint optimization. *Computer Graphics Forum 43*, 3 (2024), e15096. doi:10.1111/cgf.15096. 3

[Hua07] HUANG W.: Using eye tracking to investigate graph layout effects. In *APVIS 2007, Sydney, Australia, 5-7 February 2007* (2007), Hong S., Ma K., (Eds.), IEEE Computer Society, pp. 97–100. doi:10.1109/APVIS.2007.329282. 1, 3

[HW98] HOULE M. E., WEBBER R.: Approximation algorithms for finding best viewpoints. In *Graph Drawing* (Berlin, Heidelberg, 1998), vol. 1547 of *LNCS*, Springer Berlin Heidelberg, p. 210–223. doi:10.1007/3-540-37623-2_16. 2, 4

[JVHB14] JACOMY M., VENTURINI T., HEYMANN S., BASTIAN M.: Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLOS ONE 9*, 6 (Jun 2014), e98679. doi:10.1371/journal.pone.0098679. 2, 6

[KK88] KAMADA T., KAWAI S.: A simple method for computing general position in displaying three-dimensional objects. *Computer Vision, Graphics, and Image Processing 41* (1988), 43–56. doi:10.1016/0734-189X(88)90116-8. 2, 3

[KK89] KAMADA T., KAWAI S.: An algorithm for drawing general undirected graphs. *Information Processing Letters 31*, 1 (1989), 7–15. doi:10.1016/0020-0190(89)90102-6. 3

[KRM*17] KRUIGER 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 (Jun 2017), 283–294. doi:10.1111/cgf.13187. 2, 5

[LVJ05] LEE C. H., VARSHNEY A., JACOBS D. W.: Mesh saliency. *ACM Transactions on Graphics 24*, 3 (2005), 659–666. doi:10.1145/1073204.1073244. 3

[Mir18] MIRANDA L. J. V.: PySwarms, a research-toolkit for Particle Swarm Optimization in Python. *Journal of Open Source Software 3* (2018). URL: https://doi.org/10.21105/joss.00433, doi:10.21105/joss.00433. 5

[MPWK24] MOONEY G. J., PURCHASE H. C., WYBROW M., KOBOUROV S. G.: The multi-dimensional landscape of graph drawing metrics. In *IEEE PacificVis* (2024), pp. 122–131. doi:10.1109/PacificVis60374.2024.00022. 4

[MRS96] MONIEN B., RAMME F., SALMEN H.: A parallel simulated annealing algorithm for generating 3D layouts of undirected graphs. In *Graph Drawing* (Berlin, Heidelberg, 1996), Brandenburg F. J., (Ed.), vol. 1027 of *LNCS*, Springer Berlin Heidelberg, p. 396–408. doi:10.1007/BFb0021823. 2

[MSC24] MARTIN M. Y., SBERT M., CHOVER M.: Viewpoint selection for 3d-games with f-divergences. *Entropy 26*, 6 (May 2024), 464. doi:10.3390/e26060464. 2

[OTWO17] OHTAKA Y., TAKAHASHI S., WU H.-Y., OHTA N.: Using mutual information for exploring optimal light source placements. In *Proc. Intl. Symp. Smart Graphics* (2017), Springer, pp. 155–166. 4

[PB96] PLEMENOS D., BENAYADA M.: Intelligent display techniques in scene modelling. In *GraphiCon* (1996), pp. 1–5. 2, 3, 4

[PCJ96] PURCHASE H., COHEN R., JAMES M.: Validating graph drawing aesthetics. In *Graph Drawing* (Germany, 1996), Brandenburg F., (Ed.), vol. 1027 of *LNCS*, Springer Berlin Heidelberg, pp. 435–446. doi:10.1007/bfb0021827. 1, 3

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KOPF A., YANG E., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. 5

[SPFG05] SBERT M., PLEMENOS D., FEIXAS M., GONZÁLEZ F.: Viewpoint quality: Measures and applications. In *Eurographics Workshop on Computational Aesthetics in Graphics, Visualization and Imaging* (2005), The Eurographics Association Press Aire-la-Vile, pp. 185–192. 2

[Stü98] STÜRZLINGER W.: *Imaging all Visible Surfaces Or How many Reference Images are needed for Image-Based Modeling?* Technical Report TR98-010, Univ. of North Carolina at Chapel Hill, Mar 1998. 3

[TCG22] TIEZZI M., CIRAVEGNA G., GORI M.: Graph neural networks for graph drawing. *IEEE Transactions on Neural Networks and Learning Systems 35*, 4 (2022), 4668–4681. doi:10.1109/TNNLS.2022.3184967. 3

[VBP*09] VIEIRA T., BORDIGNON A., PEIXOTO A., TAVARES G., LOPES H., VELHO L., LEWINER T.: Learning good views through intelligent galleries. *Computer Graphics Forum 28*, 2 (2009), 717–726. doi:10.1111/j.1467-8659.2009.01412.x. 3

[VFSH01] VÁZQUEZ P.-P., FEIXAS M., SBERT M., HEIDRICH W.: Viewpoint selection using viewpoint entropy. In *Vison Modeling and Visualization* (2001), vol. 1, pp. 273–280. doi:/10.2312/COMPAESTH/COMPAESTH05/185-192. 2, 3

[VGO*20] VIRTANEN P., GOMMERS R., OLIPHANT T. E., HABERLAND M., REDDY T., COURNAPEAU D., BUROVSKI E., PETERSON P., WECKESSER W., BRIGHT J., VAN DER WALT S. J., BRETT M., WILSON J., MILLMAN K. J., MAYOROV N., NELSON A. R. J., JONES E., KERN R., LARSON E., CAREY C. J., POLAT İ., FENG Y., MOORE E. W., VANDERPLAS J., LAXALDE D., PERKTOLD J., CIMRMAN R., HENRIKSEN I., QUINTERO E. A., HARRIS C. R., ARCHIBALD A. M., RIBEIRO A. H., PEDREGOSA F., VAN MULBREGT P., SCIPY 1.0 CONTRIBUTORS: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods 17* (2020), 261–272. doi:10.1038/s41592-019-0686-2. 5

[vWMT24] VAN WAGENINGEN S., MCHEDLIDZE T., TELEA A.: An experimental evaluation of viewpoint-based 3d graph drawing. *Computer Graphics Forum 43*, 3 (June 2024), e15077. doi:10.1111/cgf.15077. 1, 2, 3, 4

[WF96] WARE C., FRANCK G.: Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Transactions on Graphics 15*, 2 (1996), 121–140. doi:10.1145/234972.234975. 1, 2

[WM08] WARE C., MITCHELL P.: Visualizing graphs in three dimensions. *ACM Transaction on Applied Perception 5*, 1 (jan 2008). doi:10.1145/1279640.1279642. 1, 2

[WW22] WEISS S., WESTERMANN R.: Differentiable direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 28*, 1 (2022), 562–572. doi:10.1109/TVCG.2021.3114769. 3

[WWS*17] WANG Y., WANG Y., SUN Y., ZHU L., LU K., FU C.-W.: Revisiting stress majorization as a unified framework for interactive constrained graph visualization. *IEEE TVCG 24*, 1 (2017), 489–499. 10

[WYHS21] WANG X., YEN K., HU Y., SHEN H.: DeepGD: A deep learning framework for graph drawing using GNN. *IEEE Computer Graphics and Applications 41*, 05 (2021), 32–44. doi:10.1109/MCG.2021.3093908. 5

[WZZ*07] WANG Y., ZHOU D., ZHENG Y., WANG K., YANG T.: Viewpoint selection using pso algorithms for volume rendering. In *International Multi-Symposiums on Computer and Computational Sciences* (2007), pp. 286–291. doi:10.1109/IMSCCS.2007.56. 3

[YZY18] YANYANG Z., ZEHAO Z., YUNXIA F.: Volume rendering viewpoint selection based on adaptive scaleable chaotic particle swarm optimization. *Journal of System Simulation 30*, 12 (2018). doi:DOI:10.16182/j.issn1004731x.joss.201812013. 3

[ZFY20] ZHANG Y., FEI G., YANG G.: 3d viewpoint estimation based on aesthetics. *IEEE Access 8* (2020), 108602–108621. doi:10.1109/ACCESS.2020.3001230. 3

[ZY22] ZHANG Y., YANG G.: Optimization of the virtual scene layout based on the optimal 3d viewpoint. *IEEE Access 10* (2022), 110426–110443. doi:10.1109/ACCESS.2022.3214206. 3