A PARALLELIZABLE HEURISTIC ALGORITHM FOR PLANAR TRIANGULATION WITH NOISY DATA

Boris Melnikov
Faculty of Computational Mathematics and Cybernetics
Shenzhen MSU-BIT University
China
bormel@mail.ru

Bowen Liu
Faculty of Computational Mathematics and Cybernetics
Shenzhen MSU-BIT University
China
liubowenmbu@163.com

Article history:
Received 18.09.2025, Accepted 20.11.2025

Abstract

Distance-based triangulation is a fundamental technique for localization and mapping in two-dimensional spaces. Potential applications include wireless sensor networks, mobile robotics, and mapping systems, where real-time and reliable two-dimensional localization is essential. In addition to the usual triangulation algorithms, physics also uses its development, the so-called causal dynamic triangulation, which connects quantum geometry with the concepts of space and time. In it, significant results were achieved in modeling black holes and cosmological conditions, in which conventional triangulation methods make it possible to understand what the plane of the event horizon may consist of. In real-world applications, however, sensor measurements are often corrupted by noise and unpredictable errors, making it infeasible to reconstruct the true coordinates of points directly from raw distances. Classical optimization-based approaches to restore the coordinates of the entire set of points can mitigate noise but usually incur high computational costs, as they attempt to minimize global error measures. To balance computational efficiency and reconstruction accuracy, we introduce a heuristic algorithm for planar point coordinate recovery. The method is amenable to parallel computation and is able to return solutions of acceptable quality within practical time limits. The experimental methods proposed in this paper and their corresponding estimates show that the proposed approach makes it possible to reliably reconstruct the configurations of flat points, despite noisy distance data.

Key words

triangulation, noisy distance data, planar coordinate reconstruction, heuristic algorithm, parallel computation

1 Introduction

Distance-based triangulation is a classical method for determining the positions of points in a plane from pairwise distance measurements. In addition to the usual applications of triangulation algorithms, physics also uses some of its development options, for example, the so-called causal dynamic triangulation, which connects quantum geometry with the concepts of space and time, [Smit, 2024; Shi et al., 2025] etc. In it, significant results were achieved in modeling black holes and cosmological conditions, in which conventional triangulation methods make it possible to understand what the plane of the event horizon may consist of. The main position of causal dynamic triangulation can be considered the discreteness of space-time. Simplices are "glued together" so as to preserve the causal structure (i.e., the cause precedes the effect). This is achieved by distinguishing between spatial and time-like edges in a triangulation, namely, spatial edges connect points in one time layer, and time-like edges connect points between adjacent time layers. In this case, in one of the methods of considering the corresponding models, all possible triangulations of simplices are summed up, weighted by their total action.

Let us move on to the problem and a brief description of the algorithms for solving it. In the ideal case, when measurements are exact, triangulation produces consistent and accurate coordinates. In practical settings, however, sensor data are often contaminated with noise and unpredictable errors. These inaccuracies can make the reconstructed coordinates inconsistent with the true point configuration, sometimes rendering exact recovery impossible.

Existing approaches often formulate the reconstruction problem as a global optimization task, typically aiming to minimize an error function over all points simultaneously. While these formulations can, in principle, reduce the effect of noise, they are computationally expensive and scale poorly to large instances. The high cost arises from their reliance on iterative solvers or relaxation methods that require repeated global updates.

In this work, we propose a heuristic algorithm for reconstructing planar point coordinates from noisy distance data. The algorithm is designed with parallelization in mind, enabling substantial reductions in runtime compared to sequential implementations. Instead of pursuing an exact global optimum, our approach focuses on achieving practical efficiency. The experimental section provides a detailed analysis of the runtime improvements obtained through parallelization, as well as the growth of computational cost as the number of points increases.

The remainder of this paper is organized as follows. Section 2 introduces the research background and formulates the reconstruction problem. Section 3 presents the proposed heuristic algorithm, followed by a description of its parallelization strategy. Section 4 outlines the experimental setup, while reporting and discussing the results. Finally, Section 5 concludes the paper and highlights possible directions for future work.

2 Background

The problem of reconstructing point coordinates from pairwise distances is well-studied in various fields, including sensor network localization, robotics, and computational geometry. When distance measurements are exact, classical methods such as multidimensional scaling (MDS, [Rusch, 2025] etc.) or trilateration can be employed to recover the coordinates accurately. The purpose of this method is to represent objects or observations as points in a multidimensional space, while preserving their mutual distances as accurately as possible; this practically coincides with the formulation of the triangulation problem. However, in the presence of noise, these methods may yield suboptimal or inconsistent results. Various optimization-based techniques have been proposed to address the challenges posed by noisy data. These methods typically involve minimizing a global error function, such as the sum of squared differences between measured and reconstructed distances. While effective in reducing noise impact, these approaches often require significant computational resources, especially for large datasets.

The problem of recovering point positions from distances has a long history in distance geometry [Mucherino et al., 2013; Lee et al., 2014]. When the measurements are exact, trilateration or triangulation techniques can reconstruct point configurations uniquely up to rigid transformations [Gkioulekas et al., 2024; Garamvölgyi et al., 2022]. However, in practice, distance data are often contaminated with noise, making exact reconstruction infeasible.

One of the most prominent applications is in wireless sensor network localization, where sensor positions are estimated using inter-sensor distance measurements [Paul and Sato, 2017]. Similar formulations appear in robotics for simultaneous localization and mapping (SLAM) [Torres-González et al., 2018].

To address noise, many existing methods formulate the reconstruction as a global optimization problem, often minimizing a least-squares error between measured and estimated distances [Liberti et al., 2014]. Convex relaxations, such as semidefinite programming (SDP), have been proposed to improve robustness [Biswas et al., 2006]. While these approaches provide theoretical guarantees in certain cases, they are computationally expensive and do not scale well to large datasets [Shahbazian and Ghorashi, 2017].

As datasets grow larger and sensor networks become denser, the computational cost of global optimization becomes a bottleneck. This motivates the development of heuristic and parallelizable methods that trade exact optimality for efficiency and scalability [Shang et al., 2003]. Such methods aim to produce approximate yet robust reconstructions within practical runtime limits, making them suitable for large-scale applications.

3 Proposed Method

3.1 Problem Description

We study the task of reconstructing a set of n points in the Euclidean plane $\mathbb{R}^2$ from noisy pairwise distance measurements. Let the unknown correct positions be

$$X^* = \{x_1^*, x_2^*, \dots, x_n^*\}, \quad x_i^* \in \mathbb{R}^2.$$

The true Euclidean distance between two points i and j is

$$d_{ij} = \|x_i^* - x_j^*\|_2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm.

In practice, the observed measurements are corrupted by *multiplicative noise*, modeled as

$$\tilde{d}_{ij} = \eta_{ij}\, d_{ij},$$

where η_{ij} represents a relative error term. This formulation captures scenarios where measurement noise scales with the magnitude of the true distance.

Collecting all noisy observations yields a symmetric matrix

$$\widetilde{D} = \left(\tilde{d}_{ij}\right)_{i,j=1}^{n} \in \mathbb{R}^{n\times n},$$

which we call the *observed distance matrix*. The objective is to reconstruct an estimated configuration

$$\hat{X} = \{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_n\}, \quad \hat{x}_i \in \mathbb{R}^2,$$

such that the pairwise Euclidean distances $\|\hat{x}_i - \hat{x}_j\|_2$ agree with $\widetilde{D}$ as closely as possible.

Because Euclidean distances are invariant under rigid transformations, the reconstruction $\hat{X}$ is only unique up to global translation, rotation, and reflection. Thus, the fundamental problem can be stated as follows:

Given a noisy Euclidean distance matrix $\widetilde{D}$ corrupted by multiplicative errors, reconstruct the point configuration in $\mathbb{R}^2$ whose pairwise Euclidean distances approximate $\widetilde{D}$ as accurately as possible.

3.2 An Informal Description of the Noisy Euclidean Distance Matrix Reconstruction Algorithm

In the simplest case, we know the distance of the point to be placed (point E in the following Fig. 1) from some given ones; the number of given points is at least 3, in the figure given they are A, B, and C. In this case, all the auxiliary circles constructed (there are $2n$ circles for n points under consideration) intersect at one point; as already mentioned, in our example this is point E. At the same time, unnecessary intersections (in the figure, these are points G, H, and I) are easily filtered out by a simple auxiliary algorithm and are not used in further calculations.

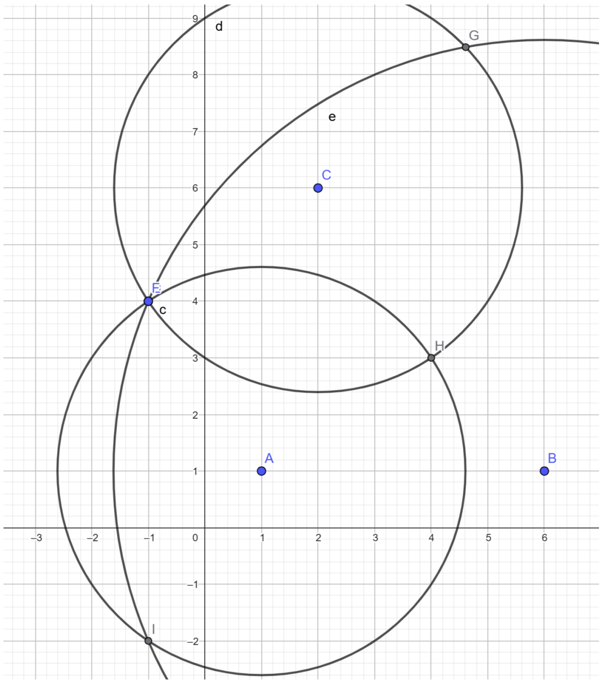

Figure 1. The simplest case of restoring the points

(The first two figures in this paper are made using the system `GeoGebra`, see [Blažek and Pech, 2025] etc.)

However, in the more complex case, which is considered in this paper, the points of the pairwise intersection of the circles do not coincide. The total number of such points is already calculated using a simple combinatorial formula, and note that even in the case of 99 points (this is the maximum dimension we are considering), the number of such points is not very large: there are 3 for 3 points, 6 for 4 points, 10 for 5 points, and so on. As in the simplest case, unnecessary intersections of circles are easily found using simple auxiliary algorithms and are not further resolved.

In the following Fig. 2, the starting points (A, B, and C) are the same, and the distances from them are almost the same. However, we get discrepancies between the three intersection points D, F, and H (we repeat that the extra intersection points are easily found and discarded by a simple auxiliary algorithm), which in our case is solved by averaging the coordinates of these points.

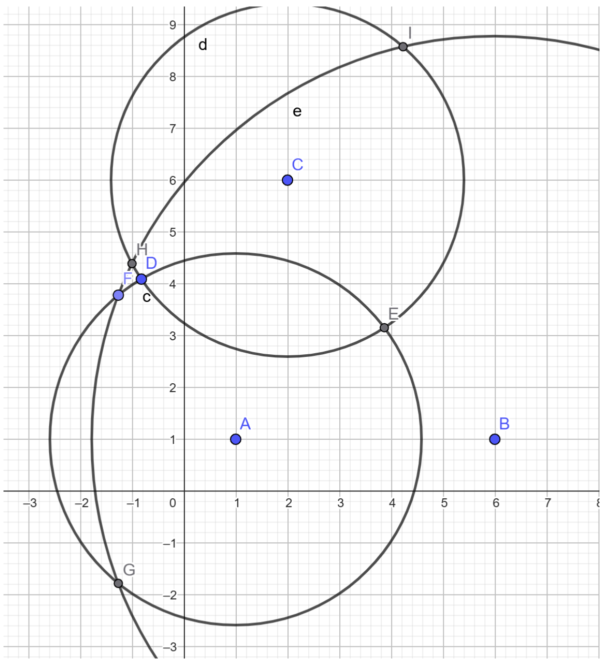

Figure 2. The usual case of restoring the points

(Compare [Beck and Sabach, 2015, Fig. 1].)

3.3 Reconstruction from a Noisy Euclidean Distance Matrix

Initialization. If $D_{ij} = 0$ for all i, j, then all points are coincident and we may set $p_i = (0, 0)$ for every i. Otherwise, select any pair (r, s) with $d_{rs} > 0$ and fix a reference frame by placing

$$p_r = (0, 0), \qquad p_s = (d_{rs}, 0).$$

This choice removes the rigid-motion ambiguity and provides the initial two restored points.

iterative point reconstruction. To determine which points can be restored in the current iteration, consider all unreconstructed points and select the best point among them as the current point to restore (the criterion for "best" is described in the Final step).

For each remaining point p_i with $i \geq 3$, the reconstruction proceeds as follows:

1. Consider **all pairs** of already restored points p_j and p_k with $j < k$ such that
$$D[j,k] > 0.$$
2. For each such pair, compute the distances
$$d_1 = D[i,j], \qquad d_2 = D[i,k], \qquad d_{jk} = D[j,k].$$
3. Apply Steps 1–4 (Collinearity check, Positioning on the line, Triangle reconstruction, and Classification of the Candidate Points) for each candidate position generated from every pair.

Step 1. Collinearity check. Check whether d_1, d_2, d_{jk} can form a triangle. If the triangle inequality fails (e.g. $d_1 + d_2 \le d_{jk}$, or similar), then p_i, p_j, p_k are collinear.

Step 2. Positioning on the line. If collinear, determine the position of p_i as follows:

Between p_j and p_k: If $d_1 + d_2 \approx d_{jk}$, adjust numerical error
$$\text{Error} = (d_1 + d_2) - d_{jk}, d_1' = d_1 - \tfrac{\text{Error}}{2},$$
and compute
$$p_i = p_j + \frac{d_1'}{d_{jk}}(p_k - p_j).$$

Beyond p_k (on ray $j \to k$): If $d_1 \ge d_2 + d_{jk}$,
$$E = \frac{d_1 - (d_2 + d_{jk})}{2}, p_i = p_j + \frac{d_1 - E}{d_{jk}}(p_k - p_j).$$

Beyond p_j (on ray $k \to j$): If $d_2 \ge d_1 + d_{jk}$,
$$E = \frac{d_2 - (d_1 + d_{jk})}{2}, p_i = p_k + \frac{d_2 - E}{d_{jk}}(p_j - p_k).$$

Step 3. Triangle reconstruction. If d_1, d_2, d_{jk} satisfy the triangle inequality, then p_i, p_j, p_k form a triangle. In this case, p_i has two possible symmetric positions with respect to the line $p_j p_k$:

1. Compute the projection of p_i onto the line segment $p_j p_k$.
2. Place p_i at distance h (the triangle height) above or below this line, obtaining two candidates:
$$p_{i,1} \quad \text{(above)}, \qquad p_{i,2} \quad \text{(below)}.$$

Step 4: Classification of the candidate points

For each new candidate point p_i, determine its position by evaluating its effect on the class bounding regions. Assume the two candidate positions are $p_{i,1}$ and $p_{i,2}$, and the existing classified points are in sets class1 and class2.

Classification process:

Special case (Collinearity):

- If, according to **Step 1. Collinearity check**, the triple p_i, p_j, p_k is collinear, then compute the single position of p_i using the rules in **Step 2. Positioning on the line** (the “between” or “beyond” formulas, including any numerical adjustment terms such as Error or E).
- In this collinear case assign the reconstructed point simultaneously to both classes:
$$\text{class1} \leftarrow \text{class1} \cup \{p_i\}, \text{class2} \leftarrow \text{class2} \cup \{p_i\}.$$

General case (Non-collinear):

- For each candidate $p_{i,\ell}$ with $\ell \in \{1,2\}$ form the minimal axis-aligned bounding rectangles
$$R_{1,\ell} = R\left(\text{class1} \cup \{p_{i,\ell}\}\right),$$
$$R_{2,\ell} = R\left(\text{class2} \cup \{p_{i,\ell}\}\right),$$
where $R(S)$ denotes the minimal axis-aligned rectangle enclosing the point set S.
- For a rectangle R with extents $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ define its area by
$$\text{area}(R) = (x_{\max} - x_{\min})(y_{\max} - y_{\min}).$$
- Consider the four rectangles
$$R_{1,1},\ R_{2,1},\ R_{1,2},\ R_{2,2},$$
corresponding to assigning $p_{i,1}$ or $p_{i,2}$ to either class1 or class2.
- Among these four, choose the rectangle with the smallest area. Assign the candidate point to the class that yields this rectangle, and assign the other candidate point to the opposite class.
- (Tie-break) If multiple rectangles have equal minimal area, choose one of them at random.

Final step: Finalize the restored coordinates

For each point p_i that has not yet been restored, first determine its candidate rectangles as in Step 4 and select the class assignment that yields the smallest axis-aligned rectangle area. Among all such unreconstructed points, choose the point p_i whose selected rectangle has the minimal area.

Compute the geometric center of the corresponding class for this p_i using an iterative optimization algorithm (e.g. well-known Weiszfeld’s method, [Venceslau et al., 2016; Neumayer et al., 2020] etc.). This geometric center will serve as the restored coordinate for this p_i.

Repeat this procedure iteratively until all points are restored.

Geometric Center Computation (Weiszfeld's Algorithm): To compute the geometric center of a set of points $\{p_1, p_2, \ldots, p_n\} \subset \mathbb{R}^2$, the algorithm minimizes the sum of Euclidean distances from the center c to each point:

$$c = \min_{x \in \mathbb{R}^2} \sum_{i=1}^{n} \|x - p_i\|.$$

Weiszfeld's algorithm is applied to solve this problem iteratively.

Step 1. Initialization. Compute the mean of the input points as the initial guess for the center:

$$c_0 = \left(\frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n} \right),$$

where (x_i, y_i) are the coordinates of point p_i.

Step 2. Iterative refinement. For each iteration k, update the geometric center c_{k+1} as

$$c_{k+1} = \left(\frac{\sum_{i=1}^{n} \frac{x_i}{\|c_k - p_i\|}}{\sum_{i=1}^{n} \frac{1}{\|c_k - p_i\|}}, \frac{\sum_{i=1}^{n} \frac{y_i}{\|c_k - p_i\|}}{\sum_{i=1}^{n} \frac{1}{\|c_k - p_i\|}} \right),$$

where $\|c_k - p_i\|$ is the Euclidean distance between the current center c_k and point p_i.

Step 3. Convergence check. Stop the iteration if the update satisfies

$$\|c_{k+1} - c_k\| < \text{tolerance},$$

where tolerance is a small positive threshold.

Step 4. Special case handling. If the distance between the current center c_k and any point p_i becomes very small (less than a threshold ϵ), then set

$$c = p_i,$$

to avoid instability.

Step 5. Termination. If the maximum number of iterations is reached before convergence, terminate the algorithm and report non-convergence.

3.4 Algorithm Advantages and Design Considerations

The workflow of the algorithm reveals several important advantages. First, the computation for each unreconstructed point is independent of the others. Moreover, within the computation for each point, the edges considered do not depend on any specific order. This independence allows for parallel execution, enabling full utilization of available computational resources when sufficient hardware is available.

Although the selection order of edges may influence the reconstruction result, for a matrix with unpredictable errors it is impossible to determine which order is optimal. Therefore, parallel computation with random order is employed to mitigate bias.

Regarding class assignment, each candidate point is assigned to the class that results in the minimal axis-aligned rectangle area. This simple strategy is motivated by the observation that, in a matrix with unpredictable errors, the class with the smallest area is likely the one least affected by errors in the current step. Consequently, the geometric center of this class is chosen as the restored coordinate for the point in this iteration.

4 Experiments

4.1 Setup of Experiments

We consider the reconstruction task under different levels of noise. For the noise parameter, we set

$$\sigma \in \{0.00, 0.02, 0.06, 0.10, 0.14\}.$$

Here, σ is interpreted as the standard deviation of a normal error model applied multiplicatively to the true distances, with random factors drawn from $\mathcal{N}(1, \sigma^2)$ and effectively bounded in $[0, 2]$. For each noise level, we test four problem sizes with 25, 33, 49, and 99 points, resulting in a grid of 8×4 configurations. Each configuration is evaluated over 20 independent runs, and we report the average performance.

The detailed procedure for each configuration is as follows.

- Generate 4 geometric distance matrices of the given size from true point configurations.
- For each geometric matrix, create 5 noisy versions (pseudo-geometric matrices) by applying perturbations corresponding to the chosen σ, yielding $4 \times 5 = 20$ experiments.
- Apply the reconstruction algorithm to each pseudo-geometric matrix to obtain a pseudo-configuration of points.
- From each reconstructed configuration, recompute a distance matrix using the *same ordering of points as in the original configuration*, and compare it with the original geometric matrix. Their difference defines the reconstruction error.
- The error, called *badness*, is measured as

$$\text{badness}(M, \hat{M}) = \sqrt{\sum_{i,j;\, i \neq j} \left(M_{ij} - \hat{M}_{ij}\right)^2},$$

  where M is the original geometric distance matrix and $\hat{M}$ is the reconstructed one. The reported value for each configuration is the average badness over 20 runs.
- Besides, for big values σ, we also we divide the root expression by $N \cdot (N - 1)$, where N is the problem size. We present these results in separate tables.

4.2 Experimental Results

We summarize the reconstruction performance under different noise levels and problem sizes in tabular form. Each entry of the table reports the average badness score over 20 independent runs for the corresponding configuration.

Table 1. Average badness scores for different noise levels σ and problem sizes N. Each value is averaged over 20 runs.

σ	$N = 25$	$N = 33$	$N = 49$	$N = 99$
0.00	0.000038	0.000036	0.000071	0.001049
0.02	0.278785	0.443629	0.545001	0.793354
0.06	0.975505	1.326922	2.152675	4.411197
0.10	1.688135	2.937301	3.889885	10.448028
0.14	3.468232	4.295704	7.277972	16.244016

As we said before, we also divide the root expression by $N \cdot (N - 1)$, where N is the problem size. The results are in the following Table 2.

Table 2. Normalized values of two rows of the previous table.

σ	$N = 25$	$N = 33$	$N = 49$	$N = 99$
0.10	0.068918	0.090389	0.080208	0.106073
0.14	0.141590	0.132191	0.150069	0.164916

In addition to Tables 1 and 2, we include a second pair of tables (Tables 3 and 4) to show the distances between the noisy distance matrices and the corresponding original distance matrices.

Table 3. Distance between noisy distance matrices and the original geometric distance matrices. Each value is averaged over 20 runs for the corresponding configuration.

σ	$N = 25$	$N = 33$	$N = 49$	$N = 99$
0.00	0.000000	0.000000	0.000000	0.000000
0.02	0.286016	0.384108	0.558426	1.131222
0.06	0.893145	1.076393	1.702814	3.358321
0.10	1.403172	1.788003	2.781705	5.609187
0.14	2.100861	2.585962	4.017488	8.012393

Table 4. Normalized values of two rows of the previous table.

σ	$N = 25$	$N = 33$	$N = 49$	$N = 99$
0.10	0.057284	0.055022	0.057357	0.056947
0.14	0.085767	0,079578	0.082839	0.081345

4.3 Visualization of Point Reconstructions

To provide a more intuitive understanding of the reconstruction performance, we visualize two examples of original point distributions alongside their corresponding reconstructions under noise level $\sigma = 0.06$.

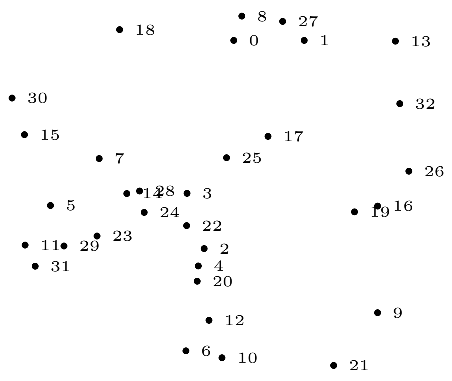

Figure 3. Reconstructed points from noisy matrix 1 ($\sigma = 0.06$)

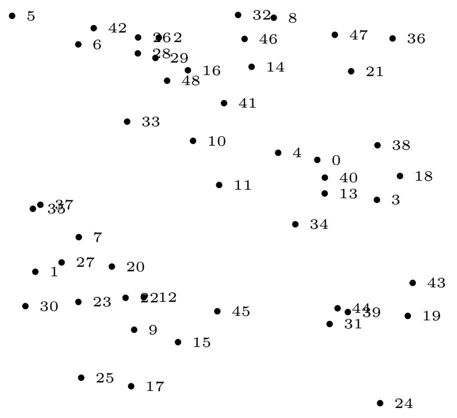

Figure 4. Original point distribution 2

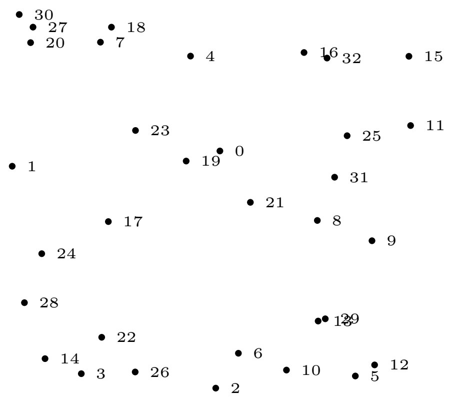

Figure 5. Original point distribution 1

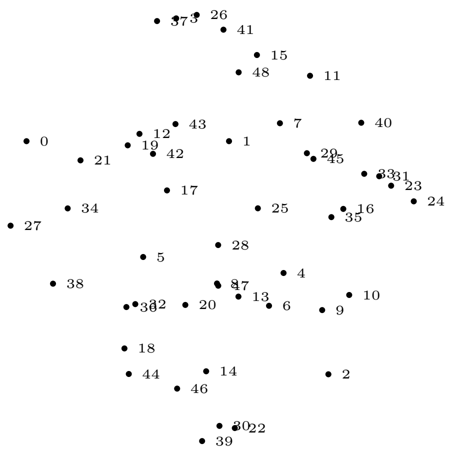

Figure 6. Reconstructed points from noisy matrix 2 ($\sigma = 0.06$)

4.4 Summary of Experimental Results

The experimental results demonstrate that our reconstruction algorithm performs reliably across different noise levels and problem sizes. In all tested configurations, the reconstruction error, as measured by the badness score, does not exceed approximately 200 % of the initial noise-induced deviation. The observed limitation is mainly due to the fact that the pseudo-noisy distance matrices cannot fully capture the true geometric structure of the original points, which inherently constrains the achievable accuracy.

In terms of computational efficiency, the algorithm exhibits strong performance. The full set of experiments for all configurations was completed in 872 seconds, indicating that the method is sufficiently fast and practical for larger problem sizes.

5 Conclusion

In this work, we have presented a heuristic algorithm for reconstructing planar point coordinates from noisy distance data. The approach effectively balances reconstruction accuracy and computational efficiency, providing reliable point placements even when the distance measurements are significantly corrupted. Experimental results demonstrate that the algorithm consistently restores point configurations with limited deviation from the true geometry, while completing all tested scenarios in a practical amount of time. This makes the method suitable for applications such as wireless sensor networks, mobile robotics, and mapping systems, where efficient and robust two-dimensional localization is critical. Future work may explore further improvements in noise tolerance and extensions to higher-dimensional reconstruction tasks.

We see that for large σ values, the normalized values are stable with increasing dimensionality, and we consider this fact as an indirect sign that our proposed algorithm is successful.

Here are some previous works by the authors of this paper, which should serve to improve the algorithm described in the article in a variety of ways:

- investigation of the random location of points from the point of view of the resulting random graph, [Melnikov and Liu, 2025];
- application of original methods for calculating rank correlation, [Melnikov and Lysak, 2024];
- the use of alternative options for averaging the obtained values (different from calculating the arithmetic mean), [Melnikov, 2024];
- using settings for previous calculation results with little change in input data, [Granichin et al., 2021] etc.;
- application of the obtained results to the creation and improvement of practical heuristic algorithms for solving the pseudogeometric version of the traveling salesman problem, [Melnikov and Chaikovskii, 2024; Melnikov et al., 2006b; Melnikov et al., 2006a; Melnikov and Terentyeva, 2025].

References

Beck, A. and Sabach, S. (2015). Weiszfeld's method: Old and new results. *Journal of Optimization Theory and Applications*, **164** (1), pp. 1 – 40.

Biswas, P., Lian, T.-C., Wang, T.-C., and Ye, Y. (2006). Semidefinite programming based algorithms for sensor network localization. *ACM Trans. Sen. Netw.*, **2** (2), pp. 188–220.

Blažek, J. and Pech, P. (2025). Solving a problem with geogebra current possibilities and limits of cas tools. *Journal of Automated Reasoning*, **69** (3).

Garamvölgyi, D., Gortler, S. J., and Jordán, T. (2022). Globally rigid graphs are fully reconstructible. *Forum of Mathematics, Sigma*, **10**, pp. e51.

Gkioulekas, I., Gortler, S. J., Theran, L., and Zickler, T. (2024). Trilateration using unlabeled path or loop lengths. *Discrete & Computational Geometry*, **71** (2), pp. 399–441.

Granichin, O., Erofeeva, V., Ivanskiy, Y., and Jiang, Y. (2021). Simultaneous perturbation stochastic approximation-based consensus for tracking under unknown-but-bounded disturbances. *IEEE Transactions on Automatic Control*, **66** (8), pp. 3710–3717.

Lee, J., Kim, Y., Lee, J., and Kim, S. (2014). An efficient three-dimensional localization scheme using trilateration in wireless sensor networks. *IEEE Communications Letters*, **18** (9), pp. 1591–1594. Publisher Copyright: © 2014 IEEE.

Liberti, L., Lavor, C., Maculan, N., and Mucherino, A. (2014). Euclidean distance geometry and applications. *SIAM Review*, **56** (1), pp. 3–69.

Melnikov, B. (2024). New algorithms for restoring dna matrix and their statistical study. *Cybernetics and Physics*, **13** (4), pp. 288 – 295.

Melnikov, B. and Chaikovskii, D. (2024). Pseudogeometric version of the traveling salesman problem, its application in quantum physics models and some heuristic algorithms for its solution. In *Springer Proceedings in Mathematics and Statistics*, vol. 446, p. 391 – 401.

Melnikov, B. and Liu, B. (2025). Special generation of random graphs and statistical study of some of their invariants. *Mathematics*, **13** (12).

Melnikov, B. and Lysak, T. (2024). On some algorithms for comparing models of femtosecond laser radiation propagation in a medium with gold nanorods. *Cybernetics and Physics*, **13** (3), pp. 261 – 267.

Melnikov, B., Radionov, A., and Gumayunov, V. (2006a). Some special heuristics for discrete optimization problems. In *ICEIS 2006 - 8th International Conference on Enterprise Information Systems, Proceedings*, vol. AIDSS, p. 360 – 364.

Melnikov, B., Radionov, A., Moseev, A., and Melnikova, E. (2006b). Some specific heuristics for situation clustering problems. In *ICSOFT 2006 – 1st International Conference on Software and Data Technologies Proceedings*, vol. 2, p. 272 – 279.

Melnikov, B. and Terentyeva, Y. (2025). Mathematical modeling of increasing the level of safety using the traveling salesman problem. *Frontiers in Artificial Intelligence and Applications*, **404**, pp. 801 – 808.

Mucherino, A., Lavor, C., Liberti, L., and Maculan, N., editors (2013). *Distance Geometry: Theory, Methods, and Applications*. Springer, New York, NY.

Neumayer, S., Nimmer, M., Setzer, S., and Steidl, G. (2020). On the robust pca and weiszfeld's algorithm. *Applied Mathematics and Optimization*, **82** (3), pp. 1017 – 1048.

Paul, A. K. and Sato, T. (2017). Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges. *Journal of Sensor and Actuator Networks*, **6** (4).

Rusch, T. (2025). Multidimensional scaling with heaviside weighting: Extensions to curvilinear component analysis and curvilinear distance analysis. *Stat*, **14** (3).

Shahbazian, R. and Ghorashi, S. A. (2017). Localization of distributed wireless sensor networks using two sage sdp optimization. *International Journal of Electrical and Computer Engineering (IJECE)*, **7** (3), pp. 1255–1261.

Shang, Y., Ruml, W., Zhang, Y., and Fromherz, M. P. J. (2003). Localization from mere connectivity. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc '03, New York, NY, USA, Association for Computing Machinery, p. 201–212.

Shi, X., Zhao, W., Chen, T., Yang, C., and Du, J. (2025). Evidence triangulator: using large language models to extract and synthesize causal evidence across study designs. *Nature Communications*, **16** (1).

Smit, J. (2024). Using massless fields for observing black hole features in the collapsed phase of euclidean dynamical triangulations. *Physical Review D*, **110** (12).

Torres-González, A., Martínez-de Dios, J. R., and Ollero, A. (2018). Range-only slam for robot-sensor network cooperation. *Autonomous Robots*, **42** (3), pp. 649–663.

Venceslau, H. M., Karam Venceslau, M. B., Xavier, A. E., and Maculan, N. (2016). A geometric perspective of the weiszfeld algorithm for solving the fermat-weber problem. *RAIRO - Operations Research*, **50** (1), pp. 157 – 173.