A Linear-Time Online Simplification Algorithm for Polylines and Polygons using a Sliding-Window Area Threshold

Hossein Narimani Rad[1], Parham Pahlavani[*2]

[1] School of Surveying and Geospatial Engineering, College of Engineering, University of Tehran, Iran – h.narimani.rad@ut.ac.ir
[2] School of Surveying and Geospatial Engineering, College of Engineering, University of Tehran, Iran – pahlavani@ut.ac.ir

Keywords: Spatial Algorithm, Generalization, Simplification, Point reduction

Abstract

This paper introduces the Cumulative Triangle Routine (CTR), a novel linear-time algorithm for simplification of polylines and polygons. CTR is designed as an online algorithm, processing points sequentially without requiring the complete geometry to be available in advance. So it can be used for real-time stream of data received from various sensors. The algorithm employs a cumulative areal threshold approach to preserve important points while achieving proper data reduction without introducing sever visual distortion. We present an evaluation comparing CTR against three established simplification methods: Ramer-Douglas-Peucker (RDP), Visvalingam-Whyatt (VW), and Normal Opening Window (NOPW). Experimental results demonstrate CTR's superior computational efficiency, maintaining consistent O(n) performance across datasets ranging from hundreds to millions of points. Qualitative analysis reveals that CTR produces visual outputs comparable to VW, particularly for man-made structures and right-angle features, while avoiding the angular distortions characteristic of RDP. Key advantages of CTR include: (1) linear-time complexity, (2) the ability to produce visually comparable results to established simplification algorithms, and (3) support for online processing of streaming data. The algorithm is useful for web-based mapping, real-time visualization, and large-scale spatial data applications where both performance and visual fidelity are essential.

1. Introduction

Simplification algorithms have been widely employed as point reduction mechanism in various applications, including cartography, digitization, multi-scale databases, visualization, and trajectories (Sun et al., 2016). These algorithms have played a crucial role in data preprocessing, enabling the generation of multi-scale representations and facilitating the preparation of geometric data for Level of Detail 0 (LOD0) in Building Information Modelling (BIM) (Pepe et al., 2024).

Numerous spatial simplification algorithms have been developed from the 1960s to the 2020s (Reumann and Witkam, 1974, Visvalingam and Whyatt, 1993, Pallero, 2013). Among these, the most prominent is the algorithm proposed by Douglas and Peucker (Douglas and Peucker, 1973), which is similar to the algorithm proposed by Ramer a year before (Ramer, 1972). Although many simplification techniques originated decades ago, recent studies have demonstrated that research in this field remains active (Chen et al., 2025, Jiang et al., 2023, Du et al., 2022, Kronenfeld et al., 2020, Liu et al., 2020).

Most existing simplification algorithms exhibit time complexities greater than linear, and the few that do operate in linear time often suffer from density sensitivity, meaning their performance depends heavily on the spatial distribution of points along the geometry. For example, the *Nth-point* algorithm, which retains every *n*-th point, may preserve too many points in dense regions while discarding critical points in sparse regions (Ekdemir, 2011). Therefore, the criterion used to retain or remove points must be independent of point density to ensure consistent simplification quality across varying geometric structures.

Fast linear-time algorithms, particularly those capable of processing data incrementally (online algorithms), offer significant advantages in applications requiring real-time performance, such as on-the-fly simplification in spatial visualization. Unlike offline methods, which require the entire geometry upfront (Frentzos and Theodoridis, 2007), online algorithms can efficiently handle data streams, making them suitable for dynamic processing pipelines. In these applications—where simplified geometries serve primarily for rendering rather than precision analysis—trade-offs between output quality or topological preservation may be justified to enhance performance.

This paper proposes a novel linear-time simplification algorithm, termed the Cumulative Triangle Routine (CTR), which processes polyline or polygon points sequentially while employing a cumulative areal threshold to preserve geometrically significant points. Unlike many existing linear-time approaches, CTR is not sensitive to local point density: by accumulating area contributions over successive vertices, it ensures that retention decisions are based on geometric significance rather than point spacing alone. As an online algorithm, CTR operates without requiring the complete geometry to be available initially.

The proposed method was evaluated through comparative analysis with established simplification techniques, including the Ramer-Douglas-Peucker (RDP) (Douglas and Peucker, 1973), Visvalingam-Whyatt (VW) (Visvalingam and Whyatt, 1993), and Normal Opening Window (NOPW) (Meratnia and de By,

* Corresponding author

2004) algorithms. Performance metrics was assessed through execution time measurements and visual output comparisons. Table 1 summarizes key characteristics of these algorithms, including their online/offline nature, year of introduction, time complexity, and respective threshold parameters.

Algorithm	Year	Online/ Offline	Time Complexity	Threshold Type
RDP	1972, 1973	Offline	$O(n^2)$	Distance
VW	1993	Offline	$O(n^2)$	Area
NOPW	2004	Online	$O(n^2)$	Distance

Table 1. Properties of some well-known simplification algorithms.

The paper is organized as follows: Section 2 reviews several point selection criteria commonly used in simplification algorithms. Section 3 presents the proposed method, while Section 4 demonstrates the results and outputs of the algorithm implementation. Finally, Section 5 provides conclusions and offers suggestions for future work.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-4-W8-2025-569-2026 | © Author(s) 2026. CC BY 4.0 License.

2. Point selection criteria

In simplification, various geometric criteria determine whether a point should be retained or discarded. van Kreveld et al. (1997) established four key measures for identifying critical points in linear features: (a) angular deviation, (b) segment length, (c) perpendicular distance, and (d) areal displacement (Figure 1) (van Kreveld et al., 1997). These metrics collectively assess a point's geometric significance, ensuring the simplified output preserves the original feature's essential characteristics.

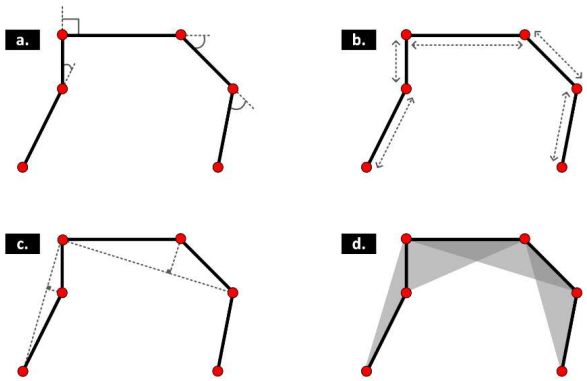

Figure 1. Geometric criteria for simplification: (a) angular deviation criterion, (b) perpendicular distance criterion, (c) vertical displacement criterion, and (d) areal displacement criterion.

The distance criterion represents one of the most fundamental approaches for identifying critical points in polyline simplification. This method eliminates vertices whose linear distance between consecutive points falls below a specified threshold. A variant employs perpendicular distance measurements, removing points that lie within a threshold distance from the line segment connecting their adjacent vertices (van Kreveld et al., 1997).

The areal displacement criterion evaluates the area formed by consecutive triplets of vertices. When this area falls below a predetermined threshold, the intermediate vertex is eliminated; otherwise, it is preserved (van Kreveld et al., 1997, Douglas and Peucker, 1973). Alternatively, the angular criterion assesses the turning angle between vectors formed by three consecutive points (van Kreveld et al., 1997). This approach retains points where the angular deviation exceeds a specified threshold, effectively preserving significant directional changes in the linear feature.

There are two distinct approaches utilizing angular change criteria. The first method employs a direction change metric, while the second implements a field of view technique. In the direction change approach, vertices exhibiting angular deviations exceeding a predetermined threshold are eliminated (denoted by grey points in Figure 2).

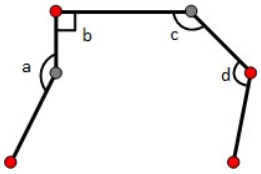

Figure 2. Removed (grey) and retained (red) points based on the angular change criterion.

The field of view criterion uses two subsequent points to build a field of view, and any following point within this field of view will be removed. Then, it uses the first point beyond this field of view to build the following field of view. Using points 1 and 2 in Figure 3, for instance, a field of view is created. As Point 3 is within the field of view, it is deleted (Figure 3.a). Point 5 falls outside the field of view and is retained (Figure 3.b). Finally, point 6 is also outside the field of view and is retained (Figure 3.c).

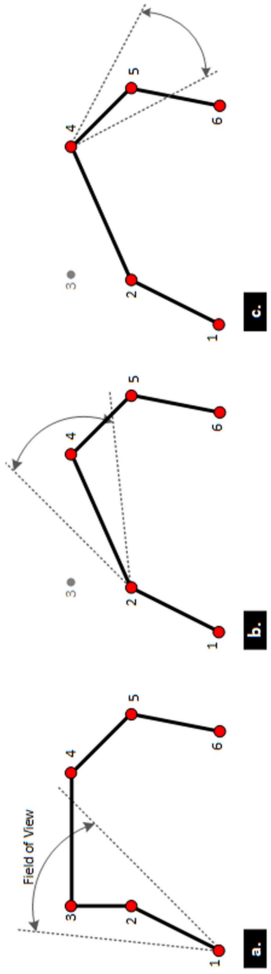

Figure 3. A simplification example using the field of view criterion.

As ongoing research in this field, Nakos and Mitropoulos proposed a more complex criterion for selecting critical points, which is based on the length ratio (LR). Their method involves calculating the intersection of a circle with a polyline or polygon for each point (Nakos and Mitropoulos, 2005). In another advancement, Jiang et al. (2023) applied deep learning techniques to automatically determine which points to retain or remove. These developments demonstrate the continued innovation in the field of simplification (Jiang et al., 2023).

3. Proposed algorithm

3.1. Algorithm's description

The proposed linear-time simplification algorithm processes a list of points by analysing successive triads of vertices from a polyline or polygon. The algorithm begins by marking the first point as retained (Figure 4.a) and computing the triangular area formed by the first three points. If this area exceeds the specified threshold, both the middle and third points are retained, and the evaluation window moves such that the third point becomes the new first point. If the area is below the threshold, it is stored and the window shifts forward by one point. For each subsequent triad, the newly computed triangle area is added to the previously accumulated value and compared against the threshold. The process continues until either the cumulative area surpasses the threshold (prompting point retention) or the endpoint is reached. Because each point is visited exactly once, the worst-case time complexity is linear, O(n). Figure 4 illustrates a sample procedure.

The proposed algorithm is presented in the following pseudocode. It accepts as input an ordered array of vertices representing either a linear feature or polygon. Through sequential processing of these vertices, the algorithm generates a simplified geometric representation.

```
Func points[] CTRSimplification(
                points[] input,
                double areaThreshold)

    points[] output;
    // add input[0] to result array
    result <= input[0];
    first = 0;
    middle = 1;
    last = 2;
    tempArea = 0;
    while last < length of input

        // Calculate the area of the current triad
        // of points
        area = GetArea(points[first],
                       points[middle],
                       points[last])
        // Check If it should retain points or not
        tempArea +=area;

        If (tempArea > areaThreshold)
            tempArea = 0;
            result <= input[middle];
            result <= input[last];
            first = last;
            middle = first + 1;
            last = first + 2;
         else
            middle++;
            last++;
    end while
    return result;
```

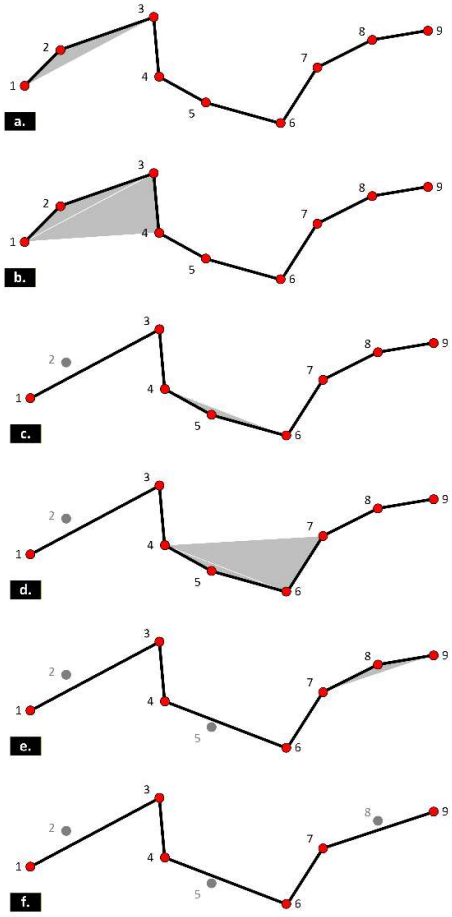

Figure 4. Simplification steps of the proposed O(n) linear-time algorithm. Grey shading indicates the cumulative area compared against the area threshold at each step, while grey markers denote removed vertices.

3.2. Proof of Time Complexity

The algorithm advances through the list of points using a single forward loop without any backtracking. For each triads of points, it performs a constant amount of work—computing the triangular area, making a small number of comparisons, and updating variables—resulting in O(1) time per step. Therefore, even in the worst case where all points are retained, each point is still visited only once, confirming that the overall time complexity remains O(n).

4. Results and Discussion

This section presents experimental evaluations of the proposed algorithm. We first examine visual simplification outputs for various feature types, followed by comparative analysis of computational performance against four established algorithms. Although a full distortion analysis is beyond the scope of this study, the Total Length of Vector Displacement (TLVD) normalized by polyline length (McMaster, 1987) is reported to provide an indicative measure of geometric deviation.

4.1. Dataset and Experimental Setup

The performance evaluation was conducted using OpenStreetMap (OSM) data of different feature types, comprising several hundred thousand features and, in total, millions of vertices. All features were projected to the Web Mercator map projection system (EPSG:3857) and simplified using comparable threshold parameters. Specifically, for each map scale, a length threshold corresponding to 4 pixels—representing a visually distinguishable change—is selected. For algorithms based on area criteria, threshold values were standardized by squaring the equivalent distance thresholds to ensure consistent levels of simplification across all methods.

4.2. Comparison of Simplification Results

To evaluate the visual quality of simplification outputs, the CTR algorithm was compared against NOPW, VW, and RDP algorithms by examining simplified features at multiple scale representations. Figure 5, Figure 6, Figure 7, and Figure 8 present comparative examples of both polygonal and linear features simplified using these methods. The visual assessment demonstrates that the proposed CTR algorithm produces competitive results, with output quality frequently resembling that of the VW.

Figure 5, demonstrates the simplification of a building polygon at approximate scales of 1:18,000, 1:9,000, and 1:4,000, with compression rates and TLVD per length provided for each algorithm. The results indicate that while the RDP algorithm achieves the highest compression, it introduce higher visual distortion. Also it tends to retain points which result in final output to be a more sharp shape, so it may not be a good choice for simplifying man-made shapes like buildings. In contrast, the proposed CTR algorithm, similar to VW, removes small-area features while better preserving right-angle geometries like buildings.

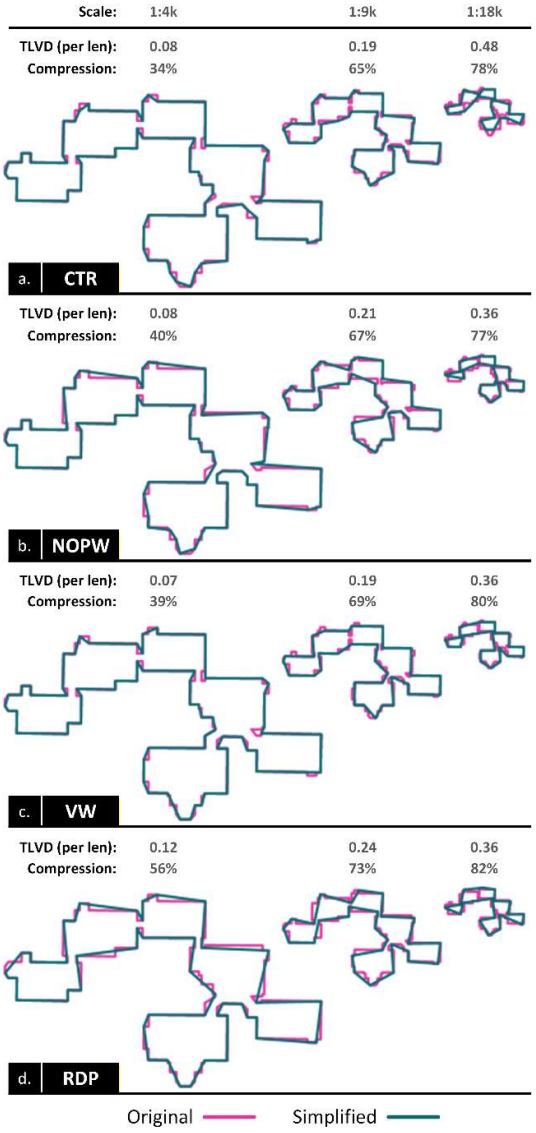

Figure 5. Polygon simplification at varying scales shows CTR and VW (area-based) remove minor features while RDP and NOPW retain small irregularities. CTR maintains shape coherence comparable to VW.

Figure 6, presents a simplified road at scales of 1:288,000, 1:144,000, and 1:72,000 using four simplification algorithms. Both VW and CTR algorithms demonstrate comparable compression rates and visual outputs. As expected, RDP achieves the highest compression rate while still maintaining acceptable visual quality for linear features.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-4-W8-2025-569-2026 | © Author(s) 2026. CC BY 4.0 License.

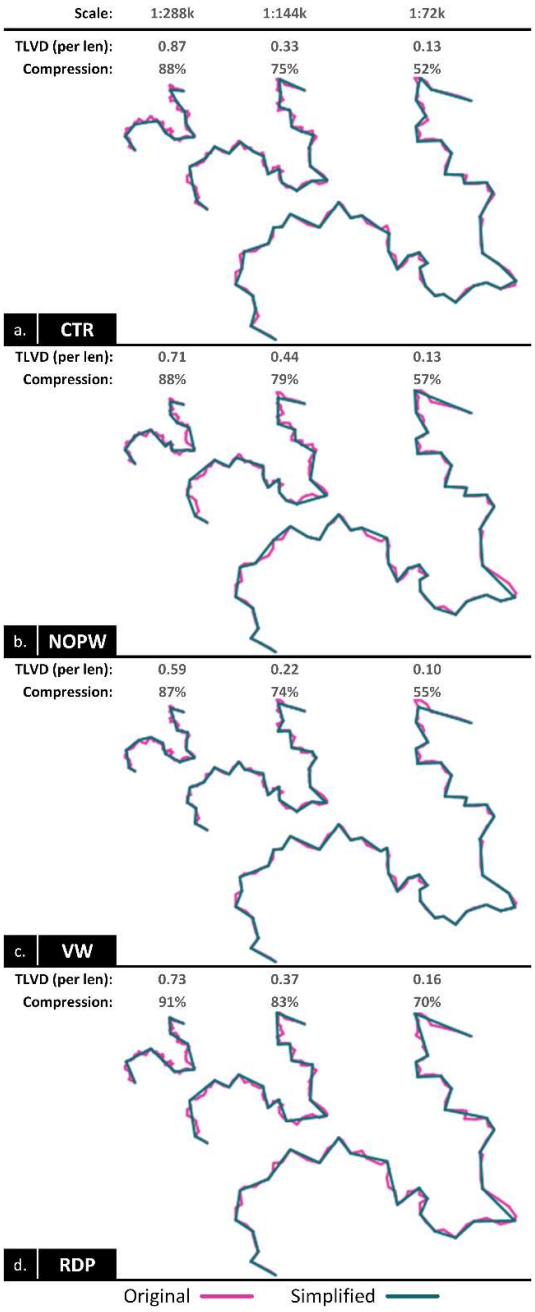

Figure 6. Simplification results of a polyline at different scales. RDP and NOPW algorithms tend to produce more rough results by retaining extreme points but VW and CTR algorithm tend to dismiss small areas providing more smooth simplified geometries.

As another example, Figure 7 presents an administrative boundary simplified at scales of 1:18,489,000, 1:9,244,000, and 1:4,622,000. Consistent with previous findings, the CTR algorithm achieves compression rates comparable to VW. Visually, both CTR and VW produce similar results - for example, both eliminate the small area at the shape's base at 1:18,489,000 scale, while NOPW and RDP retain this feature.

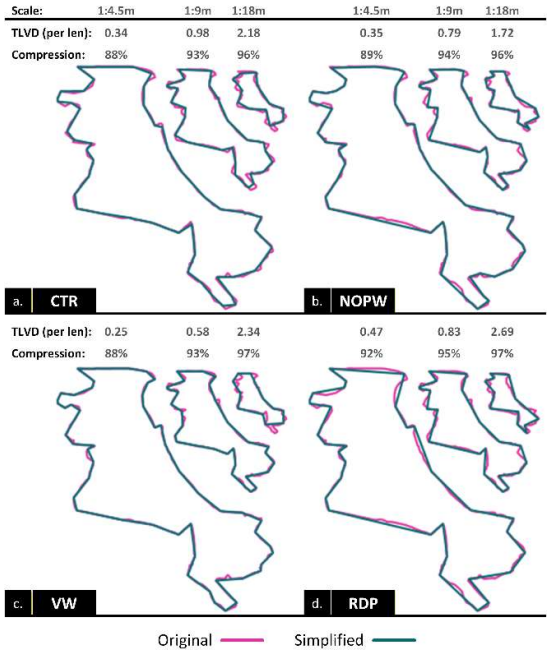

Figure 7. Sample simplification results for a polygon at three scales. CTR algorithms provides similar results to VW in the case of compression rate and visual output. NOPW and RDP also provide similar results which are higher compression rate and resulting in visually sharp shapes.

As more samples, Figure 8 presents additional simplified geometries comparing all four algorithms. The RDP algorithm typically produces acute angles and sharp features, while NOPW generates similarly angular results. In contrast, both VW and CTR create smoother approximations that better follow the original shape's median positions (Figure 8.b & c). However this totally depends on the original shape. For example Figure 8.d demonstrates an exceptional case where RDP yields the most simplified output while preserving extreme points.

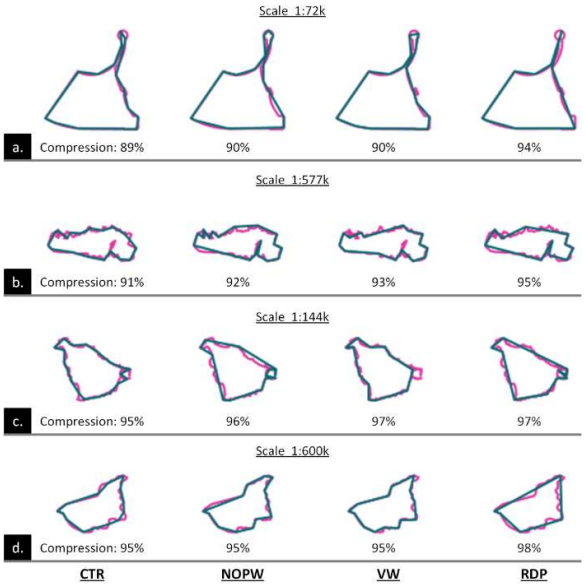

Figure 8. Simplification results by CTR, NOPW, VW, and RDP applied to geometries at different scales. RDP provides the most compression, but most visual distortion.

4.3. Execution time comparison

Execution times was measured for datasets ranging from hundreds to seven million points (Figure 9). The analysis reveals NOPW as the most computationally demanding algorithm, followed closely by VW. RDP demonstrates intermediate performance, while the proposed CTR algorithm maintains consistently superior efficiency with the lowest execution times across all tested datasets.

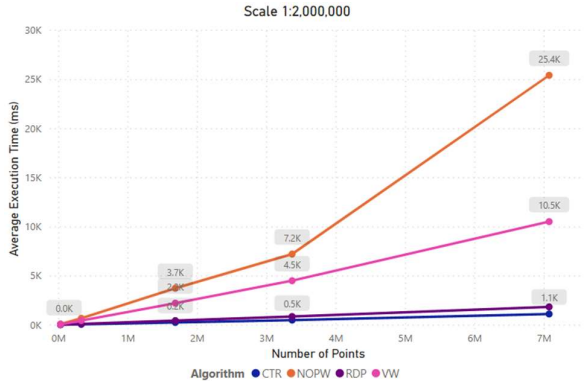

Figure 9. Execution time of algorithms simplifying at scale 1:2,000,000 when increasing the total number of points in input features.

Figure 10 compares algorithm performance at higher simplification scales (which uses lower threshold value). When processing seven million points, NOPW's execution time decreased from 25,000 ms to 15,000 ms, while VW improved from 10,000 ms to 9,000 ms. The proposed algorithm also showed a slight performance gain, reducing from 1,100 ms to 900 ms. In contrast, RDP's processing time increased from 1,100 ms to 2,100 ms. These results demonstrate CTR's consistent computational efficiency across different simplification scales. Notably, the RDP algorithm demonstrates an inverse relationship between the threshold value and computation time. In contrast to the other evaluated methods, increasing the threshold reduces processing time, as fewer intermediate points need to be examined during simplification.

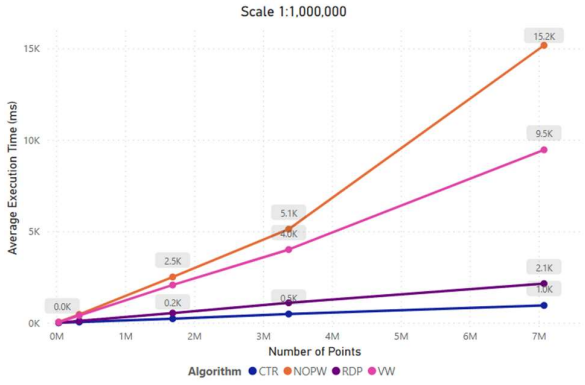

Figure 10. Execution time of algorithms simplifying at scale 1:1,000,000 when increasing number of points at higher scale which shows decrease in execution time of algorithms except the RDP algorithm (which increased from 1.1k to 2.1k compared to the Figure 9).

5. Conclusions

This paper presented the Cumulative Triangle Routine (CTR), a novel linear-time (even in worse-case) simplification algorithm for lines and polygons. Through the evaluation against established methods (RDP, VW, and NOPW), CTR demonstrated:

Computational Efficiency - Consistently achieving the lowest execution times across datasets ranging from hundreds to millions of points, while maintaining O(n) time complexity.

Visual Quality - Producing simplified outputs visually comparable to the VW algorithm, particularly effective for man-made structures and right-angle features, while providing better execution time and way simpler implementation and no need to store any extra data structures.

Practical Advantages - Operating as an online algorithm without requiring complete geometry input, making it suitable for real-time applications and streaming data scenarios.

The experimental results confirmed CTR's balanced performance in both computational speed and shape preservation. While RDP achieved higher compression rates, it introduced undesirable angular distortion. CTR's area-based approach proved particularly effective for building footprints and administrative boundaries, removing minor features while maintaining essential geometric characteristics. The algorithm's efficiency and visual quality make it particularly valuable for web-based mapping applications, real-time GIS processing, and large-scale spatial data analysis where both performance and visual distortion are important. Future work could explore a) Adaptive thresholding techniques for variable-density point clouds, b) Integration with machine learning for parameter optimization, c) Improving the visual output by trying other criteria such as cumulative perimeter.

Reference

CHEN, S., HU, A., XU, Y., WANG, H., XIE, Z. 2025: VE-GCN: A Geography-Aware Approach for Polyline Simplification in Cartographic Generalization. *ISPRS International Journal of Geo-Information,* 14**,** 64.

DOUGLAS, D. H., PEUCKER, T. K. 1973: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization,* 10**,** 112-122.

DU, J., WU, F., YIN, J., LIU, C., GONG, X. 2022: Polyline simplification based on the artificial neural network with constraints of generalization knowledge. *Cartography and Geographic Information Science,* 49**,** 313-337.

EKDEMIR, Ş. 2011: Efficient implementation of polyline simplification for large datasets and usability evaluation.

FRENTZOS, E., THEODORIDIS, Y. On the effect of trajectory compression in spatiotemporal querying. East European Conference on Advances in Databases and Information Systems, 2007: Springer, 217-233.

JIANG, B., XU, S., LI, Z. 2023: Polyline simplification using a region proposal network integrating raster and vector features. *GIScience & Remote Sensing,* 60**,** 2275427.

KRONENFELD, B. J., STANISLAWSKI, L. V., BUTTENFIELD, B. P., BROCKMEYER, T. 2020.

Simplification of polylines by segment collapse: Minimizing areal displacement while preserving area. *International Journal of Cartography,* 6**,** 22-46.

LIU, B., LIU, X., LI, D., SHI, Y., FERNANDEZ, G., WANG, Y. 2020: A Vector Line Simplification Algorithm Based on the Douglas–Peucker Algorithm, Monotonic Chains and Dichotomy. *ISPRS International Journal of Geo-Information,* 9**,** 251.

MCMASTER, R. B. 1987: The geometric properties of numerical generalization. *Geographical Analysis,* 19**,** 330-346.

MERATNIA, N., DE BY, R. A. Spatiotemporal Compression Techniques for Moving Point Objects. *In:* BERTINO, E., ed. Advances in Database Technology - EDBT 2004, 2004 Berlin, Heidelberg. Springer, 765-782.

NAKOS, B. P., MITROPOULOS, V. C. 2005: Critical point detection using the Length Ratio (LR) for line generalization. *Cartographica: The International Journal for Geographic Information and Geovisualization,* 40**,** 35-51.

PALLERO, J. L. 2013: Robust line simplification on the plane. *Computers & Geosciences,* 61**,** 152-159.

PEPE, M., PALUMBO, D., DEWEDAR, A. K. H., SPACONE, E. 2024. Toward to Combination of GIS-HBIM Models for Multiscale Representation and Management of Historic Center. *Heritage,* 7**,** 6966-6980.

RAMER, U. 1972: An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing,* 1**,** 244-256.

REUMANN, K., WITKAM, A. 1974: Optimizing Curve Segmantation in Computer Graphics. *International Comp. Sympo. 1973. Amsterdam***,** 467-472.

SUN, P., XIA, S., YUAN, G., LI, D. 2016: An overview of moving object trajectory compression algorithms. *Mathematical Problems in Engineering,* 2016.

VAN KREVELD, M., NIEVERGELT, J., ROOS, T., WIDMAYER, P. 1997: *Algorithmic foundations of geographic information systems*, Springer.

VISVALINGAM, M., WHYATT, J. D. 1993: Line generalisation by repeated elimination of points. *The Cartographic Journal,* 30**,** 46-51.