Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 6, December 2026, pp. 5681~5700
ISSN: 2302-9285, DOI: 10.11591/eei.v15i6.10300

Spatial measurement based threshold determination for polygon simplification using the Douglas-Peucker algorithm

Rahmadi Wijaya[1,2], Imas Sukaesih Sitanggang[3], Irman Hermadi[3], Annisa[2]
[1]Doctoral Program in Computer Science, School of Data Science, Mathematics and Informatics, IPB University, Bogor, Indonesia
[2]Software Engineering Application, School of Applied Science, Telkom University, Bandung, Indonesia
[3]Program in Computer Science, School of Data Science, Mathematics and Informatics, IPB University, Bogor, Indonesia

Article Info

Article history:

Received Mar 18, 2025
Revised Apr 10, 2026
Accepted Sep 12, 2026

Keywords:

Douglas-Peucker
Hotspot
Metrics
Percentage
Threshold

ABSTRACT

Forest and land fires occur almost annually in Indonesia, with hotspots serving as key indicators. Identifying high-risk areas requires visualizing hotspot distribution using polygons at the village level, covering 83,971 villages according to the Statistics Indonesia (BPS) in 2023. To enhance visualization efficiency, polygon simplification is necessary. Hotspot distribution at the village level is analyzed using an online analytical processing (OLAP) system. This study aims to determine an optimal threshold for polygon simplification. The success criteria include the number of coordinate (NuC) points, hotspot count, and GeoJSON file size. Village polygon and hotspot data were obtained from the Geospatial Information Agency (BIG) and MODIS FIRE, covering January 1, 2013 to December 31, 2023. The methodology consists of threshold determination using perimeter, area, and bounding box diagonal metrics; polygon simplification with the Douglas-Peucker (DP) algorithm; and spatial operations on hotspots and village polygons. Threshold values range from 0.05% to 0.25%. The results show that the optimal threshold is 0.15%, preserving most coordinates, maintaining hotspot count stability, and reducing the GeoJSON file size from 49.998 MB to <10.712 MB across all villages in South Sumatra Province, Indonesia.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Rahmadi Wijaya
Doctoral Program in Computer Science, School of Data Science, Mathematics and Informatics
IPB University
Bogor, Indonesia
Email: wijayarahmadi@apps.ipb.ac.id

1. INTRODUCTION

Forest and land fires are recurring environmental disasters in Indonesia, causing severe environmental, economic, and social impacts almost every year [1]–[3]. Hotspot detection based on satellite-derived thermal anomaly data has become a key indicator for early fire detection and monitoring [4]. Accurate mapping of hotspot distribution is essential for identifying high-risk areas, enabling early warning systems, and supporting informed decision-making for fire prevention and mitigation [5]. However, representing hotspot data at the village level across Indonesia, which comprises 83,971 villages according to the 2023 Statistics Indonesia (BPS) [6], poses substantial computational challenges, particularly in large-scale interactive geospatial analysis. The high complexity of polygon geometries at fine spatial resolution often results in inefficiencies in data processing, storage, and visualization, thereby limiting the performance of spatial analytics systems.

A wide range of computational and analytical approaches has been developed to improve fire monitoring and hotspot analysis, including clustering techniques such as density-based spatial clustering of applications with noise (DBSCAN)-based grouping and enhancements in satellite-based fire detection

algorithms to improve accuracy and real-time capabilities [7]–[10]. Other studies have examined the relationship between forest loss and climate impacts, emphasizing the importance of continuous spatial monitoring for environmental risk mitigation [11]–[13]. In addition, advancements in satellite constellation optimization and the integration of edge computing have been introduced to reduce latency and enhance system reliability in real-time fire detection [14]–[17]. While these developments have significantly improved detection accuracy and environmental analysis, challenges remain in efficiently managing and visualizing hotspot distributions at fine spatial units, such as villages, within large-scale geospatial systems.

Despite these advances, limitations persist in balancing computational efficiency with thematic accuracy. Prior research on Douglas-Peucker (DP) threshold optimization has primarily focused on geometric fidelity using metrics such as shape similarity and Hausdorff distance [18]–[20]. Although these metrics effectively preserve the visual structure of polygons, they often overlook thematic consistency, particularly the stability of hotspot counts after simplification. In fire risk assessment, this limitation may lead to boundary shifts and hotspot misclassification, thereby reducing analytical reliability. Furthermore, spatial efficiency aspects, such as vertex reduction and file size optimization, are rarely evaluated alongside thematic accuracy, resulting in a lack of integrated approaches that simultaneously address computational performance and data integrity [21], [22]. These limitations highlight the need for a method that integrates both spatial efficiency and thematic consistency in determining optimal simplification thresholds. To address this gap, this study contributes a novel adaptive threshold determination approach that jointly optimizes geometric fidelity, spatial efficiency, and thematic consistency, explicitly addressing the overlooked issue of hotspot stability in large-scale geospatial analysis.

Within this context, this study proposes a metric-based adaptive threshold determination approach for polygon simplification using the DP algorithm. The method integrates three geometric metrics perimeter, area, and bounding box diagonal scaled by percentage values (0.05%–0.25%) to dynamically determine acceptable simplification thresholds. By explicitly incorporating both spatial efficiency and thematic consistency, the proposed approach aims to balance geometric preservation with data reduction. The method is evaluated using village-level polygon data and hotspot records in South Sumatra Province, considering three key criteria: reduction in coordinate points, reduction in GeoJSON file size, and stability of hotspot counts. This study aims to identify an optimal simplification threshold that supports scalable geospatial analysis while maintaining data integrity and improving the reliability of fire monitoring systems. The findings are expected to enhance the efficiency and reliability of monitoring platforms, reduce computational overhead in large-scale geospatial analytics, and minimize information loss in critical decision-making processes. Furthermore, the proposed approach provides a foundation for developing adaptive and dynamic threshold determination methods that can be extended to broader spatial data generalization contexts, contributing to future research on scalable geospatial analytics and intelligent disaster management systems.

The remainder of this paper is organized as follows. Section 2 describes the materials and methods, including data sources, metric definitions, and the proposed threshold determination algorithm. Section 3 presents and discusses the experimental results, highlighting the trade-offs between data simplification and data integrity. Section 4 concludes the paper with key findings, implications for large-scale geospatial applications, and directions for future research.

2. METHOD

This section presents a comprehensive methodology for determining the optimal polygon simplification threshold using the DP algorithm [21]–[26]. The proposed approach aims to reduce computational complexity in visualizing village-level hotspot data while maintaining thematic accuracy, as defined in the research objective. The overall framework follows a structured and reproducible workflow that integrates data acquisition, preprocessing, metric-based threshold determination, polygon simplification, and performance evaluation.

Figure 1 illustrates the workflow, which begins with the acquisition of village boundary polygons in GeoJSON format from the Geospatial Information Agency (BIG) and hotspot data in CSV format from the NASA FIRMS MODIS archive. The data preprocessing stage involves converting the hotspot dataset into a geospatial layer and performing a spatial join to associate each hotspot with its corresponding village polygon. Following this integration, key geometric metrics perimeter (P), area (A), and bounding box diagonal (D) are computed and used to derive adaptive simplification thresholds expressed as percentage values (0.05%-0.25%). These thresholds dynamically adjust to the geometric characteristics of each polygon, enabling a more flexible and context-aware simplification process compared to conventional fixed-threshold approaches.

Subsequently, the DP algorithm is applied using the metric-based adaptive thresholds to generate simplified polygon geometries while preserving essential spatial and thematic properties. The performance of the simplification process is evaluated using three key indicators: reduction in coordinate points, consistency

of hotspot counts, and reduction in GeoJSON file size. These criteria ensure that the proposed method achieves a balance between geometric efficiency and thematic reliability.

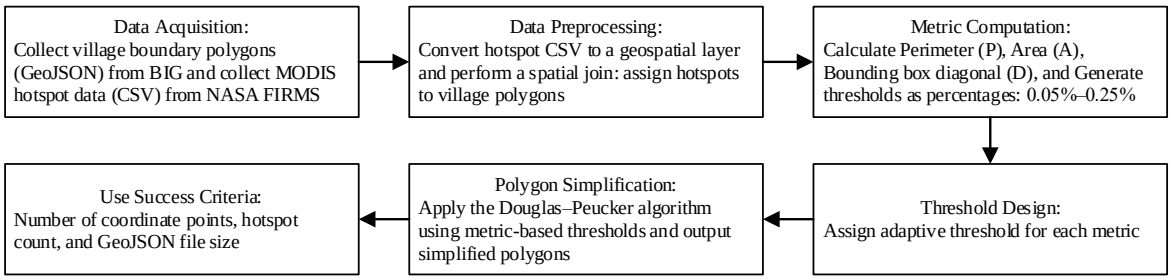


Figure 1. Workflow of the proposed methodology

2.1. Data acquisition and preprocessing

This study utilized two primary datasets. Village administrative boundary data in GeoJSON format were obtained from the BIG [27], while hotspot data were derived from the moderate resolution imaging spectroradiometer (MODIS) active fire product provided by NASA FIRMS, covering the period from January 1, 2013 to December 31, 2023 [28]. The MODIS dataset contains fire detection points with associated attributes such as acquisition time, confidence level, and brightness temperature, whereas the administrative dataset represents polygonal boundaries of villages.

Prior to analysis, both datasets underwent preprocessing to ensure spatial and attribute consistency. All data were reprojected into the WGS 84 coordinate system (EPSG:4326) using GeoPandas to maintain interoperability across layers. The administrative boundary data were validated for topological correctness by removing duplicated geometries and repairing invalid polygons following the guidelines in [29], [30]. Attribute normalization was performed to harmonize field names and ensure compatibility during spatial operations [31], [32].

The MODIS hotspot dataset was filtered to include only fire pixels with a confidence level of at least 80%, following standard practices in fire monitoring studies [33]–[35]. Each hotspot point was spatially joined to the corresponding village polygon using a point-in-polygon operation implemented in GeoPandas. This ensured that hotspot counts could be aggregated at the village level for subsequent analyses [36], [37]. Missing or null geometries, if present, were excluded from the dataset to prevent computational inconsistencies.

To facilitate reproducibility, all preprocessing steps were fully automated using Python scripts. These scripts employed libraries such as GeoPandas for spatial operations, Shapely for geometric manipulation, and Pandas for attribute handling [29]. The complete codebase and sample datasets are provided as supplementary materials, enabling independent replication of the data preparation workflow. Additionally, the workflow for processing MODIS data adheres to NASA FIRMS standard protocols [NASA FIRMS, 2023], while polygon integrity checks and reprojection steps follow established geospatial processing guidelines [29], [38].

Figure 2 presents the workflow of data acquisition and preprocessing, which involves multiple stages to ensure both spatial and thematic consistency prior to polygon simplification. The process begins with the collection of village boundary data in GeoJSON format obtained from the BIG, alongside the collection of MODIS hotspot data from the NASA FIRMS archive covering the period 2013–2023. Both datasets are then reprojected into the EPSG:4326 coordinate system (WGS 84) to ensure spatial uniformity and compatibility across sources. Once the projection is standardized, the next step is to validate polygons and fix topological errors using the GeoPandas library. This ensures that the village boundaries are free from geometric inconsistencies such as overlaps, gaps, or invalid shapes. After topology correction, the attributes of the datasets are normalized through field harmonization, enabling consistent schema alignment between spatial and non-spatial attributes. Subsequently, hotspot records are filtered by applying a confidence threshold of at least 80% to ensure that only reliable fire detections are considered for further analysis [33], [39]. Following attribute normalization and filtering, the workflow proceeds with the spatial join operation, where hotspots are assigned to village polygons using a point-in-polygon method. This allows each hotspot to be linked directly to its respective administrative unit. After the spatial join, the dataset is further refined by removing null or invalid geometries, ensuring the integrity of the spatial database. Finally, the cleaned and validated dataset is saved as preprocessed data, which serves as the input for the subsequent polygon simplification process. This comprehensive preprocessing workflow guarantees that the dataset is accurate, consistent, and suitable for robust spatial analysis.

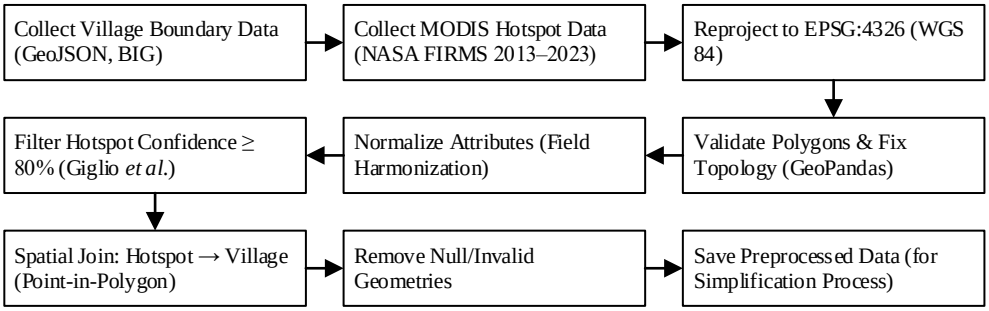


Figure 2. The workflow execution phase of data acquisition and preprocessing

Algorithm 1 illustrates that the algorithm is initiated by loading village boundary polygons in GeoJSON format and MODIS hotspot data in CSV format into the geospatial environment. The coordinate reference systems of both datasets are standardized to WGS 84 (EPSG:4326) before further processing. Next, the village polygons are validated by removing duplicate features and repairing invalid geometries, which eliminates potential topological errors. After validation, attribute normalization is performed by standardizing field names, ensuring uniformity across datasets. The hotspot data are then filtered to retain only records with a confidence level of at least 80%, thereby improving data reliability. Following this, a spatial join operation using the point-in-polygon method assigns each hotspot to its corresponding village polygon. The integrated dataset is further refined by removing records with null or invalid geometries, ensuring the integrity of the spatial data. Finally, the cleaned and integrated dataset is exported for use in subsequent analysis.

Algorithm 1. Execution phase of data acquisition and preprocessing

```
Step 1: Start process
Step 2:    Input village polygons (GeoJSON), hotspot data (CSV)
Step 3:    Load datasets and reproject to WGS 84 (EPSG:4326)
Step 4:    Clean data:
              Remove duplicates
              Repair invalid geometries
              Normalize attributes
Step 5:    Filter hotspots (confidence ≥ 80%)
Step 6:    Perform spatial join (point-in-polygon)
Step 7:    Remove invalid/null records
Step 8:    Export cleaned dataset
Step 9: End process
```

2.2. Metric computation and threshold design

A metric-based approach was adopted to overcome limitations in existing studies, which often apply static thresholds without considering thematic consistency [23], [40]. For each village polygon, the three geometric metrics were computed using the Python libraries Shapely and GeoPandas. These metrics were selected because they collectively represent linear, areal, and spatial extent characteristics, whereas alternative metrics such as curvature are difficult to integrate with thematic evaluation [18], [20]. Candidate thresholds were calculated as percentages of the original metric values, ranging from 0.05% to 0.25% in increments of 0.05%. For example:

$$Threshold_P = P \times \partial, \partial \in \{0.0005, 0.001, 0.0015, 0.0025\}$$

This study employs a combination of standard and novel methodological strategies to ensure both correctness and innovation. The standard approach involves the application of the DP algorithm, a widely recognized technique for polygon simplification that effectively reduces vertex density while preserving geometric integrity [41]. In contrast, the novelty of this research lies in the development of a metric-based adaptive threshold determination framework. This framework integrates three spatial metrics perimeter, area, and bounding box diagonal scaled proportionally to the original polygon dimensions. Unlike conventional static thresholding methods, this adaptive design ensures scalability across heterogeneous polygon geometries and balances spatial efficiency (file size and vertex reduction) with thematic consistency (hotspot stability), thereby addressing key limitations identified in prior studies.

Scaling thresholds based on polygon metrics ensures adaptability to varying shapes and sizes, addressing the trade-off between computational efficiency and thematic integrity. This adaptive thresholding provides a more robust solution compared to traditional fixed approaches [26].

The selection of multiple geometric metrics is theoretically grounded in their complementary representations of polygon characteristics. Their combined use provides a broader geometric basis for adaptive threshold determination than relying on a single metric. The mathematical formulations are given in (1) [42]–[44]:

$$perimeter = \sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \quad (1)$$

where: n represents the NuC points forming the polygon, (x_i, y_i) is the i-th coordinate point, and (x_{n+1}, y_{i+1}) considered as the first point (x_i, y_i) to close the polygon (therefore $x_{n+1} = x_i$ and $y_{n+1} = y_1$).

Algorithm 2 illustrates that the algorithm is the CalculatePolygonPerimeter algorithm computes the total boundary length of a polygon based on its vertices. The process begins by initializing the perimeter value to zero and determining the total number of vertices. Each vertex is then processed sequentially in a loop, where a pair of consecutive points is selected to form an edge. For every edge, the horizontal and vertical differences between the two points are calculated, and the Euclidean distance formula is applied to obtain the edge length. This distance is added cumulatively to the perimeter value. The loop continues until all edges are processed, including the closing edge that connects the last vertex back to the first. Finally, the algorithm returns the computed perimeter as the output.

Algorithm 2. Calculate_polygon_length (vertices)

```
Step 1: Start process
Step 2:    Input vertices = [(x₁, y₁), …, (xₙ, yₙ)]
Step 3:    Initialize perimeter = 0
Step 4:    For each vertex i:
             a. Let p₁ = (xᵢ, yᵢ) and p₂ = (xᵢ₊₁, yᵢ₊₁) (last connects to first)
             b. Compute distance: √((x₂ − x₁)² + (y₂ − y₁)²)
             c. Add to perimeter
Step 5:    Return perimeter
Step 6: End process
```

$$area = \frac{1}{2} \left| \sum_{i=1}^{n} (x_i \times y_{i+1} - y_i \times x_{i+1}) \right| \quad (2)$$

Algorithm 3 illustrates that the algorithm for computing polygon area begins by initializing the area value to zero and determining the number of vertices. Each vertex is then processed sequentially in a loop, where pairs of consecutive points are taken to compute cross-products between their coordinates. These cross-products are accumulated to form a running total that represents twice the signed area. After all vertices have been processed, including the final edge that connects the last vertex back to the first, the accumulated value is divided by two and its absolute value is taken to yield the polygon's area. The algorithm then returns this area as the final output.

Algorithm 3. Calculate_polygon_area (vertices)

```
Step 1: Start process
Step 2:    Input vertices = [(x₁, y₁), …, (xₙ, yₙ)]
Step 3:    Initialize area = 0
Step 4:    For each vertex i:
             a. Let (xᵢ, yᵢ) and next vertex (xᵢ₊₁, yᵢ₊₁) (last connects to first)
             b. Update: area += (xᵢ · yᵢ₊₁ − xᵢ₊₁ · yᵢ)
Step 5:    Compute area = |area| / 2
Step 6:    Return area
Step 7: End process
```

$$Diagonal = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2} \quad (3)$$

where: $(x_{min,}\, y_{\min})$ is the lower-left corner point of the bounding box and $(x_{max,} y_{max})$ is the upper-right corner point of the bounding box.

$$D_{BB} = \sqrt{(bounds[2] - bounds[0])^2 + (bounds[3] - bounds[1])^2} \quad (4)$$

Algorithm 4 illustrates that the algorithm for calculating a polygon's bounding box begins by extracting all x and y coordinate values from the list of vertices. From these values, the minimum and maximum of the x-coordinates are determined to define the horizontal limits, while the minimum and maximum of the y-coordinates are obtained to define the vertical limits. These four values—min_x, min_y, max_x, and max_y—are then combined to form the bounding box, which represents the smallest rectangle that completely encloses the polygon. Finally, the algorithm returns this bounding box as the output.

Algorithm 4. Calculate_polygon_ bounding box diagonal (vertices)

```
Step 1: Start process
Step 2:   Input vertices[]
Step 3:   Extract x and y coordinates
Step 4:   Compute:
            min_x, max_x from x-coordinates
            min_y, max_y from y-coordinates
Step 5:   Construct bounding box (min_x, min_y, max_x, max_y)
Step 6:   Return bounding box
Step 7: End process
```

The threshold determination process based on perimeter, area, and bounding box diagonal calculations uses percentage values ranging from 0.05% to 0.25%, with an increment (rk) of 0.05. Threshold values obtained from percentages below 0.05 do not significantly reduce the NuC in the simplified polygon. In contrast, threshold values from percentages greater than 0.25 result in a substantial reduction in the NuC, causing the simplified polygon to no longer resemble the original shape. Using a rk value below 0.05 produces nearly identical threshold values for each metric. Conversely, using a rk value greater than 0.05 results in different threshold values for each metric, with varying decimal precision across the metrics.

In this study, these metrics were selected for threshold determination because they capture the global geometric characteristics of polygons in a computationally efficient manner. Unlike curvature or vertex density, which emphasize local shape details and are sensitive to noise and data resolution, global metrics such as perimeter and area provide a stable representation of polygon scale and complexity, while the bounding box diagonal serves as a straightforward proxy for spatial extent. These metrics are particularly suitable for large-scale datasets, such as village-level administrative boundaries, where simplification must balance computational efficiency with geometric fidelity. Previous research has highlighted that global descriptors are effective for maintaining spatial identity in generalization tasks [26], [32], and bounding measures have been widely adopted to approximate polygon complexity in large geospatial databases [45]–[47]. Therefore, the adoption of perimeter, area, and bounding box diagonal ensures both theoretical relevance and practical feasibility in preserving thematic consistency during simplification, while also allowing for adaptive threshold computation across diverse polygonal geometries.

To further strengthen the methodological rationale, it is important to highlight the applied advantages of the selected metrics. Perimeter, area, and bounding box diagonal can be computed quickly even for very large datasets, making them highly suitable for nationwide applications involving hundreds of thousands of polygons. Their global nature ensures that the simplification process remains consistent across diverse geometries, while at the same time avoiding the computational overhead associated with local descriptors such as curvature or vertex density. Moreover, these three metrics are sufficiently representative to preserve the consistency of hotspot counts before and after simplification, which is essential for ensuring reliable fire monitoring and spatial analysis outcomes.

In addition, although alternative metrics such as curvature and vertex density are commonly employed for capturing fine-scale geometric details, their suitability is limited when applied to large-scale spatial datasets [48]. Curvature-based descriptors emphasize local changes in shape, making them effective for tasks such as feature recognition or the analysis of intricate polygonal outlines [48], [49]. Similarly, vertex density can provide insights into localized complexity by quantifying the concentration of coordinate points within specific segments of a polygon [50]. However, both approaches are computationally intensive and highly sensitive to noise and data resolution, which significantly reduces their efficiency in large-scale applications involving hundreds of thousands of polygons, such as nationwide administrative boundaries. By contrast, global metrics such as perimeter, area, and bounding box diagonal provide a more scalable and robust solution, enabling efficient threshold computation while preserving overall geometric fidelity [45], [46], [51]. This comparative rationale further supports the adoption of global descriptors in the present study to balance computational feasibility and thematic consistency at scale.

Algorithm 5 illustrates that the algorithm for determining the threshold value begins with a polygon defined by its vertices, a percentage factor (α), and a chosen metric type. If the selected metric is perimeter, the boundary length is computed using the Euclidean distance formula. If the metric is area, the polygon area is calculated with the Shoelace formula, while for the diagonal metric, the length of the bounding box diagonal is determined. Once the metric value is obtained, it is multiplied by the percentage factor to generate the adaptive threshold. Finally, this threshold value is returned as the output.

Algorithm 5. Threshold calculation based on selected metric (simplified)

```
Step 1: Start process
Step 2:   Input polygon P, scaling factor α, metric_type ∈ {Perimeter, Area, Diagonal}
Step 3:   Compute metric value:
```

```
            Perimeter → Euclidean distance;
            Area → Shoelace formula; Diagonal → bounding box diagonal
Step 4:    Calculate threshold = metric × α
Step 5:    Return threshold value
Step 6: End process
```

This algorithm calculates the threshold dynamically based on the selected geometric metric and user-defined percentage factor. The flexibility to select different metrics enables experimentation with multiple threshold strategies.

2.3. Polygon simplification using the Douglas-Peucker algorithm

Polygon simplification was performed using the DP algorithm, implemented through the simplify() function in the GeoPandas library, leveraging the precomputed metric-based thresholds described in subsection 2.2. For each candidate threshold, the algorithm generated a simplified polygon layer by iteratively removing vertices that deviated from the baseline by less than the specified tolerance value (ε) [23], [40], [52], while retaining critical points to maintain geometric fidelity. This approach enables substantial reduction in geometric complexity without compromising the structural characteristics of the original polygons.

Figure 3 illustrates the step-by-step process of the DP algorithm, depicted sequentially from Figures 3(a) to (h) [52]. In Figure 3(a), the algorithm begins by selecting the first and last vertices of the polyline, namely points a and h. A base segment (a–h) is drawn, and all intermediate vertices (b–g) are evaluated with respect to their perpendicular distance from this segment. Point c emerges as the farthest vertex, exceeding the tolerance (ε). Figure 3(b) highlights this calculation, where the perpendicular distance from c to segment a–h is greater than ε, thus c must be retained. Consequently, the polyline is divided into two subsegments, a–c and c–h, to be processed recursively. Figure 3(c), the subsegment a–c is evaluated. The only intermediate point, b, has a perpendicular distance larger than ε, so it is retained, and the subsegments a–b and b–c are finalized. Figure 3(d) presents the evaluation of subsegment c–h, which contains vertices d, e, f, and g. Among them, g is identified as the farthest point, and since its distance to segment c–h exceeds ε, it is preserved, resulting in two new subsegments: c–g and g–h. Figure 3(e) shows the recursive process on subsegment c–g. The intermediate vertices (d, e, f) are examined, and point d is identified as the farthest from segment c–g. Since its distance is greater than ε, d is retained, forming subsegments c–d and d–g. Figure 3(f) illustrates the evaluation of subsegment g–h, where the only intermediate point, f, exceeds the threshold and is therefore preserved. Next, in Figure 3(g), the subsegment d–g is considered with e as the intermediate vertex. The perpendicular distance of e to segment d–g falls within the tolerance band (≤ε), leading to the elimination of e. Finally, Figure 3(h) presents the simplified polyline obtained after the recursive procedure is completed. The retained vertices are a, b, c, d, f, g, and h, while point e is removed. This result demonstrates the principle of the DP algorithm: vertices that contribute significantly to the geometric fidelity of the polyline are preserved, while others are discarded if they lie within the tolerance threshold ε.

The DP algorithm is widely recognized for its efficiency in reducing vertex density without compromising geometric fidelity, making it suitable for large-scale spatial datasets. In this study, the DP tolerance (ε) was not a fixed value [53], but dynamically computed using adaptive thresholds derived from perimeter, area, and bounding box diagonal metrics, scaled by percentage factors (0.05%–0.25%).

For example, consider a village polygon with a perimeter of 18,547.29 units. At a threshold factor of 0.15%, the DP tolerance was computed as:

$$\varepsilon = 18{,}547.29 \times 0.0015 = 27.82$$

This value was used as the tolerance parameter in the simplify() function.

The DP algorithm begins by identifying the point with the greatest perpendicular distance from the baseline formed between the first and last vertices of a polygonal chain [53]. The algorithm initializes the maximum distance and corresponding index, then iteratively evaluates each intermediate point. If the largest distance exceeds the tolerance value (ε), the algorithm recursively divides the point sequence into two subsets—between the first point and the index, and between the index and the last point—before applying the same procedure. The results of the two recursive calls are then merged to form the simplified representation. Conversely, if no point exceeds the tolerance, the algorithm retains only the first and last vertices as the simplified segment. Finally, the resulting set of points is returned as the simplified polygonal chain.

Algorithm 6 illustrates that the DP algorithm begins by identifying the point with the greatest perpendicular distance from the baseline formed between the first and last vertices of a polygonal chain. The algorithm initializes the maximum distance and corresponding index, then iteratively evaluates each intermediate point. If the largest distance exceeds the tolerance value (ε), the algorithm recursively divides the point sequence into two subsets-between the first point and the index, and between the index and the last point-before applying the same procedure. The results of the two recursive calls are then merged to form the

simplified representation. Conversely, if no point exceeds the tolerance, the algorithm retains only the first and last vertices as the simplified segment. Finally, the resulting set of points is returned as the simplified polygonal chain.

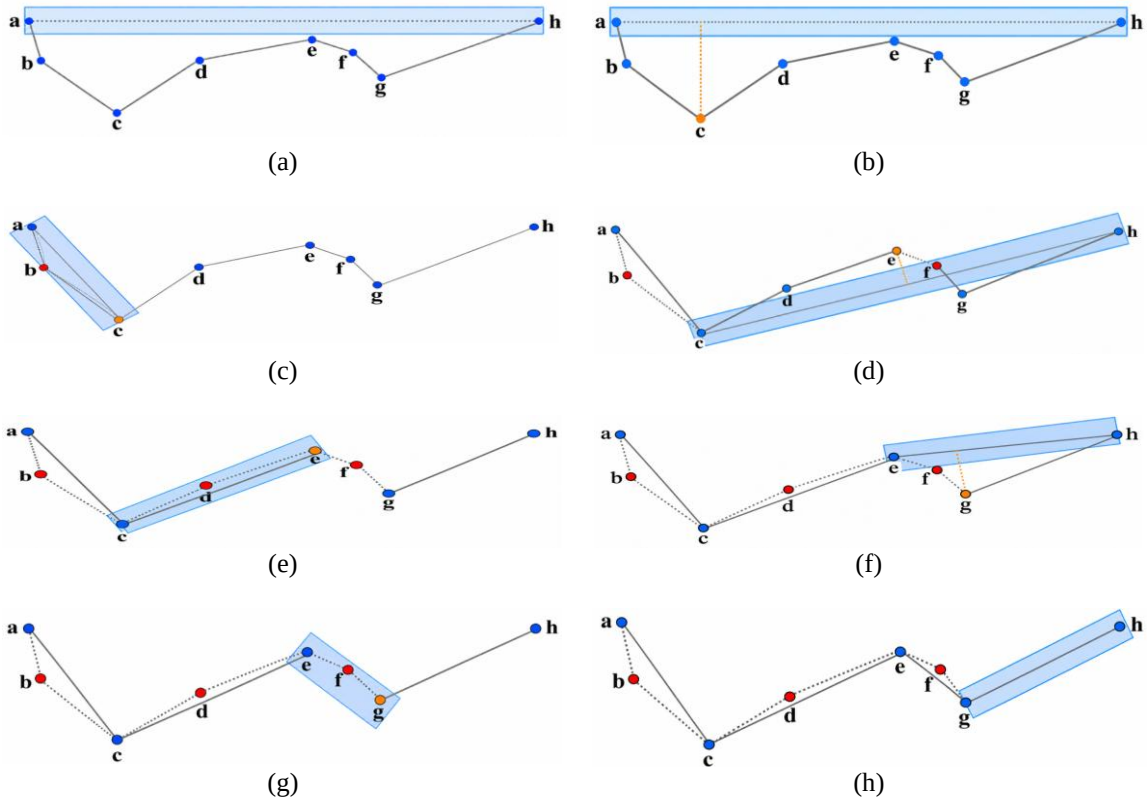


Figure 3. Step-by-step polygon simplification using the DP algorithm: (a) initial line segment (a-h); (b) identification of vertex c exceeding tolerance (ε); (c) retention of vertex b in segment a-c; (d) retention of vertex g in segment c-h; (e) retention of vertex d in segment c-g; (f) retention of vertex f in segment g-h; (g) removal of vertex e in segment d-g; and (h) final simplified polyline retaining vertices a, b, c, d, f, g, and h [52]

Algorithm 6. DP algorithm for polyline simplification

```
Step 1: Start process
Step 2:    Input set of points, tolerance ε
Step 3:    Find point with maximum perpendicular distance (dmax) to line (first-last point)
Step 4:    If dmax > ε:
              a. Recursively process two segments (first-farthest, farthest-last)
              b. Combine results
Step 5:    Else: Keep only first and last points
Step 6:    Return simplified points
Step 7: End process
```

The DP algorithm was chosen due to its proven capability to achieve significant vertex reduction while preserving geometric integrity [54], making it well-suited for handling large-scale GeoJSON datasets used in real-time geospatial visualization and disaster monitoring applications [21], [55].

The DP algorithm was chosen over alternatives such as Visvalingam–Whyatt due to its superior ability to preserve geometric fidelity under varying tolerance levels and its proven computational efficiency in large-scale applications [56]–[58], as supported by previous cartographic generalization studies. Algorithm 7 shows algorithm for determining the threshold value based on metrics using a polygon.

Algorithm 7 describes a process that loads the shapefile into a GeoDataFrame and initializes threshold storage for perimeter, area, and bounding box. For each valid polygon, the algorithm computes these metrics and derives thresholds by multiplying them (perimeter, √area, and diagonal) with the factor (α). The polygon is then simplified using each threshold. After all polygons are processed, the average threshold for each metric is calculated and returned as the final output.

Algorithm 7. Metric-based polygon simplification with adaptive thresholds

```
Step 1: Start process
Step 2:   Input shape_file, α
Step 3:   Load shapefile into GeoDataFrame
Step 4:   Initialize threshold storage {perimeter, area, bounding_box}
Step 5:   For each Polygon:
           a. Compute perimeter, area, and bounding box diagonal
           b. Calculate thresholds:
                Perimeter × α; √Area × α; Diagonal × α
           c. Store threshold values
           d. Simplify polygon using each threshold
Step 6:   Compute average threshold for each metric
Step 7:   Output average thresholds (perimeter, area, bounding box)
Step 8: End process
```

All computations and the DP algorithm were implemented in Python using libraries such as GeoPandas for spatial data handling, Shapely for geometric operations, and NumPy for numerical calculations [29]. The complete source code, along with sample datasets and execution instructions, is provided in the supplementary materials to ensure full reproducibility.

2.4. Use success criteria

The evaluation stage was designed to validate whether the polygon simplification process described in subsection 2.3, based on thresholds computed in subsection 2.2, by considering three key criteria: NuC points, GeoJSON file size, and hotspot count, to ensure computational efficiency and analytical reliability.

2.4.1. Polygon coordinate count calculation method

To measure the geometric complexity of each polygon in the dataset, the first step is to calculate the NuC points (vertices) that define the polygon geometry. This coordinate count serves as an indicator of the spatial detail preserved after the simplification process, making it an important metric for evaluating simplification quality.

In principle, the coordinate count for a single polygon is calculated based on the total number of points forming its exterior boundary (exterior ring). In cases where the geometry type is MultiPolygon, the calculation is performed by summing all coordinate points from each individual polygon that composes the geometry. Mathematically, the NuC in the simplified multipolygon can be expressed as (5) [53]:

$$C_i = \sum_{j=1}^{k_i} \left| coords\left(P_{i,j}\right) \right| \tag{5}$$

where: k_i is the number of polygons in the i-th multipolygon and $P_{i,j}$ is the j-th polygon in the ith multipolygon, The NuC on the polygon boundary line. Thus, the total NuC from all polygons in the dataset:

$$N = \sum_{i=1}^{M} C_i = \sum_{i=1}^{M} \sum_{j=1}^{k_i} \left| coords\left(P_{i,i}\right) \right| \tag{6}$$

with the assumption that the polygon has k_i=1, whereas the multipolygon has k_i>1, N is the total NuC from all polygons in the dataset, and M is the NuC in the dataset. Algorithm 8 shows algorithm polygon coordinate counting.

Algorithm 8 illustrates that the algorithm begins by checking whether the specified directory is valid; if not, the process is terminated with an error message. Once validated, an empty list is initialized to store the results. The algorithm then iterates through each file in the directory and processes only those with a ".shp" extension. For every shapefile, the data are read into a GeoDataFrame, and each row is examined to ensure geometric validity. If the geometry type is a polygon, the NuC in the exterior boundary is counted and recorded. For multipolygon geometries, the algorithm iterates through each polygon component, counts the coordinates of its exterior boundary, and appends the results to the list. Finally, the procedure returns a collection containing the coordinate counts for all polygons across the processed shapefiles.

Algoritma 8. Polygon coordinate counting

```
Step 1: Start process
Step 2:   Input directory
Step 3:   Validate directory (stop if invalid)
Step 4:   Initialize results list
Step 5:   For each .shp file in directory:
           a. Load as GeoDataFrame
           b. For each record:
                Check valid geometry
                If Polygon
                   count exterior coordinates
```

```
                If MultiPolygon
                   iterate each polygon and count coordinates
            c. Store {file_name, index, coordinate_count}
Step 6:   Output all coordinate counts
Step 7: End process
```

2.4.2. Conversion of Shapefile to GeoJSON and file size calculation

This stage aims to measure the impact of polygon simplification on spatial data storage efficiency. The process involves converting the dataset from Shapefile (.shp) format to GeoJSON (.geojson) to facilitate its use in web-based applications and online analytical processing (OLAP) systems, followed by calculating the size of the converted GeoJSON files as an indicator of storage efficiency

The GeoJSON format was chosen because it supports text-based spatial data representation, which can be easily integrated with analytical systems and web-based services. After conversion, the file size is measured in Bytes (B) and then converted into Kilobytes (KB) and Megabytes (MB) to align with standard data efficiency analysis. The GeoJSON file size of the multipolygon is determined by the NuC in the simplified shapefile [59]:

$$S = \sum_{i=1}^{M} C_i \times B + O \tag{7}$$

where: S is the GeoJSON file size, M is the total number of polygons in the dataset, C_i is the NuC in the i-th polygon, B is the average storage size per coordinate, and O is the metadata overhead of the GeoJSON format. Thus, the total GeoJSON file size is calculated from all polygons in the dataset:

$$S_{KB} = \frac{(\sum_{i=1}^{M} C_i \times B) + O}{1024} \tag{8}$$

Algorithm 9 illustrates that the algorithm begins by checking whether the target directory for GeoJSON files exists; if not, it creates the directory. It then iterates through all files in the input directory and processes only those with a ".shp" extension. Each shapefile is read into a GeoDataFrame and subsequently converted and saved as a GeoJSON file in the designated output directory. After conversion, an empty list is initialized to record the file information. The algorithm then iterates through each GeoJSON file, verifies its extension, and retrieves the file size in bytes. The size is also converted into kilobytes and megabytes for easier interpretation. Finally, the filename along with its size in bytes, kilobytes, and megabytes is appended to the results list, providing a structured summary of the output files.

Algorithm 9. Convert Shapefile to GeoJSON and compute file size

```
Step 1: Start process
Step 2:   Input dir_shapefile, dir_geojson
Step 3:   Ensure dir_geojson exists (create if not)
Step 4:   For each .shp file in dir_shapefile:
            a. Load as GeoDataFrame
            b. Convert and save as GeoJSON in dir_geojson
Step 5:   For each .geojson file in dir_geojson:
            a. Get file size (bytes)
            b. Convert to KB and MB
            c. Store results {file_name, size_bytes, KB, MB}
Step 6:   Output all file size results
Step 7: End process
```

2.4.3. Hotspot count

To evaluate the impact of polygon simplification on both thematic accuracy and storage efficiency. Hotspot Count per Village Polygon representing the number of detected fire hotspots within each simplified village boundary. This metric measures thematic consistency after simplification. For each village polygon, the number of hotspots (NoH) intersecting the polygon area was computed using spatial join operations. The mathematical formulation is: the NoH in the simplified multipolygon is expressed as [60], [61]:

$$H = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \delta(h_{i,j}, D_i) \tag{9}$$

where: H is the total NoH within the multipolygon, M is the total number of village polygons in the dataset, N_i is the total NoH in the dataset that are located within village polygons, D_i, $h_{i,j}$ is the i-th hotspot within the i-th polygon, and $\delta(h_{i,j}, D_i)$ is indicator function (equal to 1 if $h_{i,j}$ located within a village polygon D_i, and 0 otherwise). Indicator function $\delta(h_{i,j}, D_i)$ defined as (10):

$$\delta(h_{i,j}, D_i) = \begin{cases} 1, \ if \ h_{i,j} \in D_i \\ 0, \ if \ h_{i,j} \notin D_i \end{cases} \quad (10)$$

Filtering based on a threshold, where the presence of the 'value' attribute in the village polygon dataset is checked. If the attribute exists, hotspots with values greater than or equal to the threshold are retained:

$$H' - \{h_k | \ value(h_k) \geq threshold\} \quad (11)$$

The filtered hotspots are checked for their presence within village polygons using the 'within' operator:

$$h_k \in D_i \leftrightarrow geometry(h_k) \subseteq geometry(D_i) \quad (12)$$

Thus, obtaining the total NoH (H) within all village polygons.

$$H = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \delta(h_{i,j}, D_i) \quad (13)$$

The total NoH (H) is determined by filtering hotspots based on a threshold and verifying whether each hotspot is located within a village polygon.

Algorithm 10 illustrates that the algorithm is designed to integrate village boundary data, threshold parameters, and hotspot information for spatial analysis. First, the function read_csv_shapefiles processes CSV files containing village polygons by checking the existence of a geometry column, converting it into a GeoDataFrame, and storing the results in a list. Similarly, the function read_threshold_csv extracts threshold values from CSV files and maps them to each village in a dictionary structure. These two components provide the spatial and attribute inputs required for further computation. Next, the function count_hotspots_within_desa identifies hotspots that meet the specified intensity threshold. It filters the hotspot dataset based on the threshold, validates the existence of the required attribute column, and then performs a spatial query to count the number of filtered hotspots within each village boundary. This ensures that only relevant hotspots are included in the final analysis. Finally, the main function orchestrates the entire workflow by reading village shapefiles, loading thresholds, and processing the hotspot shapefile. For each village, the algorithm retrieves the corresponding threshold, counts the hotspots using the filtering and spatial containment procedure, and stores the results in a dictionary. The final output is a mapping of each village to its respective hotspot count, providing a structured basis for evaluating spatial and thematic consistency.

Algorithm 10. Hotspot counting within village polygons using threshold filtering

```
Step 1: Start process
Step 2:   Input village_directory, threshold_directory, hotspot_shapefile
Step 3:   Read village CSV files → convert to GeoDataFrames (WKT) → store in list
Step 4:   Read threshold CSV files → map {village_name: threshold}
Step 5:   Load hotspot shapefile into GeoDataFrame
Step 6:   For each village:
            a. Extract village name
            b. Get threshold (or default)
            c. Filter hotspots with value ≥ threshold
            d. Select hotspots within village polygon (spatial filter)
            e. Count hotspots
Step 7:   Store result in dictionary {village_name: hotspot_count}
Step 8: End process
```

The evaluation covered a total of 2,531 village polygons from South Sumatra Province For each polygon [27], 15 simplified variants were generated, corresponding to five threshold levels (0.05%, 0.10%, 0.15%, 0.20%, and 0.25%) applied across three metric types. This resulted in 2,531×15=37,965 polygon-variant evaluations for spatial and thematic consistency.

While this study primarily focused on evaluating vertex reduction, file size efficiency, and thematic consistency in terms of hotspot stability, the assessment of geometric fidelity was not included as part of the quantitative validation framework. Geometric accuracy metrics such as the Hausdorff distance and shape similarity index have been widely used in previous studies to measure the degree to which simplified polygons preserve the original geometry [62]–[65]. However, the main objective of this research was to ensure thematic reliability in fire monitoring applications, where the preservation of hotspot counts is a critical requirement. The inclusion of vertex count reduction and file size analysis already provides a measure of computational efficiency, while hotspot consistency confirms the preservation of spatial analysis outcomes. Future work will incorporate additional geometric fidelity metrics, including Hausdorff distance and shape similarity index, to further strengthen the validation of geometric identity preservation in large-scale polygon simplification tasks.

3. RESULTS AND DISCUSSION

Provincial boundary data for all regions in Indonesia were obtained from BIG [27]. The provincial shapefile was then split to obtain shapefiles for all villages within the respective provinces. The village shapefile format is used in this study.

This study evaluates vertex reduction, file size efficiency, and hotspot stability, as these directly impact computational performance and thematic consistency in fire monitoring. Geometric fidelity metrics (e.g., Hausdorff distance and shape similarity index) were excluded due to limitations and the focus on hotspot count stability, but will be incorporated in future work for more comprehensive validation.

In this study, the evaluation of hotspot stability was deliberately limited to the consistency of hotspot counts before and after polygon simplification, since this represents the primary thematic variable in fire monitoring applications. Detailed spatial validation, including the analysis of false positives, false negatives, and potential spatial displacement, was not incorporated, as the methodological emphasis was placed on ensuring thematic consistency rather than spatial relocation effects. This approach aligns with the research objective of confirming that polygon simplification does not alter the fundamental outcomes of hotspot-based thematic analysis. Future work will incorporate spatial validation, including the evaluation of false positives, false negatives, and potential spatial displacement [65], to further confirm that hotspot locations remain correctly assigned within polygon boundaries post-simplification.

3.1. Development of an optimal threshold determination algorithm

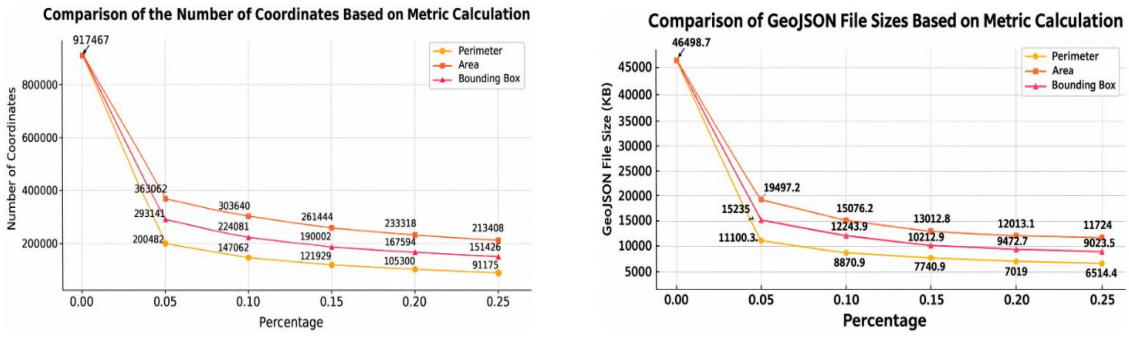
This stage begins with reading the shapefiles of all villages in South Sumatra Province, which consists of 2,531 villages [27]. The perimeter, area, and bounding box diagonal metrics are then calculated for each village shapefile. The results of these calculations are multiplied by percentage values to determine the optimal threshold for each village shapefile. The obtained optimal threshold values are then used in the village polygon simplification stage to assess the NuC, the NoH, and the GeoJSON file size. The functions involved in determining the optimal threshold include a function to calculate the threshold using metric parameters and percentage values, a function to compute the perimeter metric, a function to calculate the area metric, and a function to determine the bounding box diagonal metric.

The threshold determination process uses percentage values ranging from 0.05% to 0.25%, with an increment of 0.05%. Threshold values obtained from percentages below 0.05 do not significantly reduce the NuC in the simplified polygon. In contrast, threshold values from percentages greater than 0.25 result in a substantial reduction in the NuC, causing the simplified polygon to no longer resemble the original shape.

3.2. Simplification of all village polygons

All village polygons are simplified using their respective threshold values from the previous stage. Figure 4 compares coordinate counts across villages in South Sumatra Province under different metric-based simplification thresholds (0.05%–0.25%). The largest reduction occurs between 0% and 0.05%, followed by a gradual decline. At 0.25%, counts fall from 917,467 to 93,735 using perimeter and 213,408 using area, indicating that area-based simplification preserves more geometric detail.

At a 0.25% threshold, the bounding box metric reduces the coordinate count from 917,467 to 151,364, retaining more coordinates than the perimeter metric but fewer than the area metric (Figure 4(a)). GeoJSON file sizes decrease from 46,998.7 KB to 6,512.7 KB, 11,734 KB, and 9,025.5 KB for the perimeter, area, and bounding box metrics, respectively (Figure 4(b)). Thus, the perimeter metric achieves the greatest data reduction, while the area metric retains the most geometric detail.



(a) (b)

Figure 4. Comparison; (a) graph of the comparison of the NuC and (b) GeoJSON file size before and after simplification of all villages in South Sumatra Province

Figure 5 shows that polygon simplification reduces the Air Balui Village boundary from 91 coordinates to 36, 35, 32, and 27 at thresholds of 0.05, 0.10, 0.15, and 0.20, respectively, while retaining its overall shape. Increasing the threshold to 0.25 yields no further coordinate reduction.

Hotspot counts before and after simplification are compared to assess count preservation. Table 1 summarizes coordinate counts and GeoJSON file sizes for perimeter, area, and bounding box metrics at thresholds of 0.05–0.25%. The original 917,467 coordinates and 3,587,937,620 bytes (approximately 3.34 GiB) provide the baseline for evaluating reductions in geometric complexity and storage requirements.

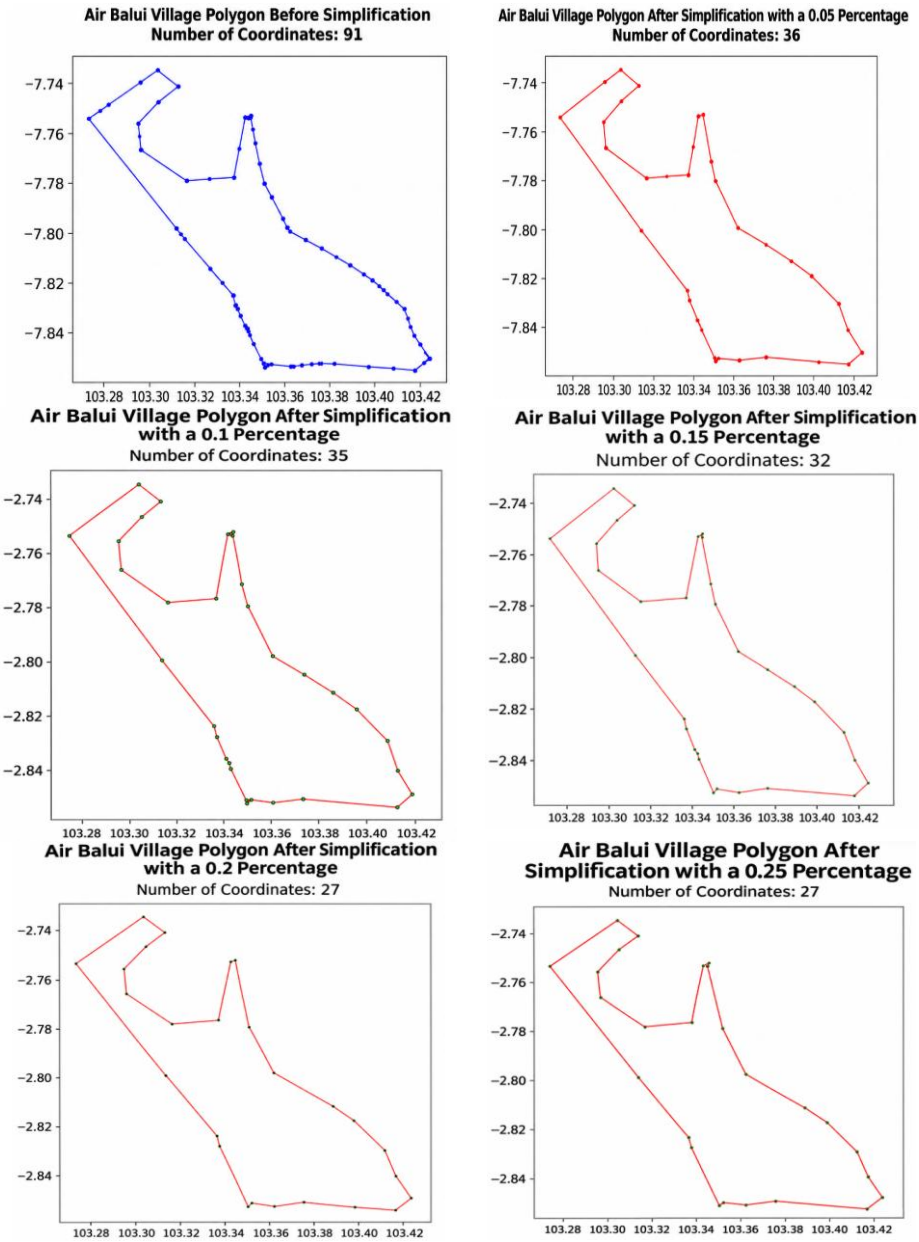


Figure 5. Number of polygon coordinates in Air Balui Village resulting from simplification using percentages of 0.05, 0.1, 0.15, 0.2, and 0.25

Based on Table 1, the NuC before simplification was 917,467. The percentage calculations of 0.05, 0.1, 0.15 (Table 1(a)), 0.2, and 0.25 (Table 1(b)) produced threshold values for all villages. The simplification results using the specified threshold parameters yielded the following NuC: 293,594 (0.05%), 224,908 (0.1%), 190,009 (0.15%), 167,563 (0.2%), and 151,364 (0.25%). The smallest reduction in the total NuC for all villages in South Sumatra Province occurred at the 0.05% threshold, resulting in 293,594 coordinates (a reduction of approximately 68%). In contrast, the highest reduction was observed at the 0.25% threshold, yielding 151,364

coordinates (a reduction of approximately 94.4%). The higher the percentage used, the greater the reduction in the NuC.

Table 1. Comparison of the NuC and GeoJSON file size from Polygon simplification of all villages in South Sumatra Province with percentage; (a) 0.05, 0.1, and 0.15 and (b) 0.2 and 0.25

(a)

No	Village	Before simplification		Percentage 0.05			Percentage 0.1		
		NuC	GeoJSON file (KB)	Threshold	NuC	GeoJSON file size (KB)	Threshold	NuC	GJSON file (KB)
1	Aceh	311	15.8427734	1.0669E-05	161	7.767578125	2.1338E-05	115	5.758789
2	Adil Makmur	69	4.15039065	5.9496E-05	43	2.685546875	0.00011893	34	2.290039
3	Air Balui	91	5.18164065	9.8892E-05	36	2.334960938	0.00019775	35	2.291016
…	…	…	…	…	…	…	…	…	…
2531	Yudha Karya Bakti	100	5.62304685	2.2057E-05	58	3.297851563	4.4115E-05	51	2.990234
	Total coordinates	917467	358793760		293594	15236.1		224908	12235.89

(b)

No	Village	Percentage 0.15			Percentage 0.2			Percentage 0.25		
		Threshold	NuC	GeoJSON file size (KB)	Threshold	NuC	GeoJSON file size (KB)	Threshold	NuC	GeoJSON file size (KB)
1	Aceh	3.20E-05	94	4.839844	4.26E-05	85	4.448242	5.33458E-05	77	4.097656
2	Adil Makmur	0.000189	29	2.071289	0.000986	24	1.852539	0.000297482	23	1.808594
3	Air Balui	0.000278	32	2.160156	0.00057	27	1.94043	0.000494463	27	1.94043
…	…	…	…	…	…	…	…	…	…	…
2531	Yudha Karya Bakti	6.61E-05	46	2.771484	8.82E-05	41	2.552734	0.000110288	39	2.464844
	Total coordinates		190009	10712.5		167563	9732.709		151364	9025.539

The size of the GeoJSON file before simplification was 3,503,105 MB or 3.421 GB. The simplification process using the specified threshold parameters resulted in GeoJSON file size of 15,236.1 MB (0.05%), 12,235.89 MB (0.1%), 10,712.5 MB (0.15%), 9,732.709 MB (0.2%), and 9,025.539 MB (0.25%). The smallest reduction in GeoJSON file size for the total villages in South Sumatra Province occurred at the 0.05 percentage threshold, resulting in a file size of 15,236.1 MB. The higher the percentage used, the greater the reduction in GeoJSON file size. The file size reduction occurred across all percentage thresholds, with an approximate reduction of 99.9%.

Figure 6 compares hotspot counts before and after polygon simplification using three geometric metrics: perimeter, area, and bounding box diagonal. Before simplification (0.00%), all three metrics show an initial hotspot count of 72,375. At a threshold of 0.05%, minor variations occur, yielding 72,382, 72,374, and 72,380 hotspots for the perimeter, area, and bounding box metrics, respectively. As the threshold increases to 0.10% and 0.15%, the perimeter-based approach shows a gradual decrease in hotspot counts, reaching 72,369 and 72,342, respectively, while the other two approaches exhibit relatively smaller variations. At higher thresholds of 0.20% and 0.25%, the perimeter-based approach records 72,336 and 72,340 hotspots, respectively. Meanwhile, the area-based approach reaches the highest count of 72,386 at 0.25%, whereas the bounding box-based approach maintains relatively stable results and returns to the original total at 0.15%. Overall, the results demonstrate that hotspot count variations depend on the selected geometric metric and simplification threshold.

To further examine hotspot count consistency at the administrative level, Table 2 presents the village-level results across 2,531 villages in South Sumatra Province. The table provides a detailed comparison of hotspot counts before and after simplification at three selected threshold levels (0.05%, 0.10%, and 0.15%), allowing variations in individual villages to be examined beyond the provincial-level results shown in Figure 6.

The total hotspot count before simplification was 72,375. After simplification, the aggregate counts were 72,380 and 72,381 at thresholds of 0.05% and 0.10%, respectively, before returning to the original total of 72,375 at 0.15%. This confirms that the 0.15% threshold preserves the overall provincial hotspot count. However, an unchanged aggregate count does not necessarily indicate that hotspot counts remain identical in

every village, as local increases and decreases may offset each other. Figure 7 therefore provides a visual example of hotspot distribution changes following polygon simplification in Danau Cala Village.

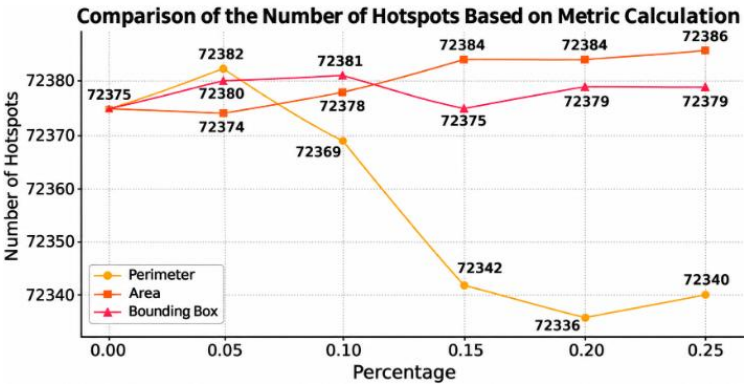


Figure 6. Graph of the comparison of the NoH before and after simplification of all Villages in South Sumatra Province

Table 2. Comparison of the NoH from polygon simplification of all villages in South Sumatra Province using percentage of 0.05, 0.1, and 0.15

No	Village	Before simplification	Percentage 0.05		Percentage 0.1		Percentage 0.15	
		NoH	Threshold	NoH	Threshold	NoH	Threshold	NoH
1	Aceh	0	1.02E-05	0	2.1E-05	0	3.2E-05	0
2	Adil Makmur	89	5.9E-05	89	0.00993	89	0.08489	89
	Air Balui	130	9.85E-05	130	0.0085	130	0.0678	130
…	…	…	…	…	…	…	…	…
2531	Yudha Karya Bakti	0	2.27E-05	0	4.2E-05	0	6.3E-05	
	Total coordinates	72375		72380		72381		72375

Figure 7 illustrates the hotspot distribution in Danau Cala Village before and after polygon simplification at different threshold levels. The first plot shows the original polygon, containing 552 hotspots, which serves as the baseline for comparison. The subsequent plots show the results at thresholds of 0.05%, 0.10%, 0.15%, 0.20%, and 0.25%. At the first three threshold levels, the hotspot count remains unchanged at 552, indicating that the simplification process preserves hotspot allocation within the village boundary. However, at thresholds of 0.20% and 0.25%, the count increases slightly to 553. This difference of one hotspot suggests that greater simplification may alter boundary precision and affect the inclusion of points located near polygon boundaries. Overall, the results illustrate that polygon simplification can reduce geometric complexity while largely maintaining hotspot allocation, although minor variations may occur at higher threshold levels.

At the provincial level, the total NoH across all village polygons in South Sumatra varies slightly with the simplification threshold: 72,380 at 0.05%, 72,381 at 0.10%, and 72,375 at 0.15% (Table 2). Compared with the original total of 72,375 hotspots, these results represent increases of five and six hotspots at the two lower thresholds, whereas the 0.15% threshold preserves the original total. This finding supports the selection of the bounding box metric with a 0.15% threshold as a suitable configuration for preserving the aggregate hotspot count while simplifying village boundaries. However, agreement at the provincial level does not necessarily imply that hotspot allocation remains unchanged within every village, as gains and losses among individual polygons may offset one another. The effectiveness of this threshold should therefore be interpreted in terms of aggregate count preservation, with village-level comparisons needed to assess spatial allocation accuracy.

We found that the adaptive threshold based on the bounding box diagonal strongly correlates with both geometric fidelity and thematic consistency, as it maintained 100% hotspot stability while reducing the GeoJSON file size by more than 99%. Specifically, the 0.15% threshold consistently produced the best trade-off between vertex reduction and hotspot preservation, demonstrating that higher percentages (≥0.20%) tend to introduce thematic deviations due to boundary shifts. This indicates that the proposed metric-based approach effectively balances spatial efficiency and data integrity compared to conventional static thresholds.

Our findings also indicate that reducing polygon complexity through adaptive thresholding is not associated with a loss of thematic consistency, despite a significant decrease in vertex count and file size. This contrasts with previous studies that primarily focused on geometric accuracy using measures such as Hausdorff distance and shape similarity [19], [20], [63] without evaluating thematic stability. By integrating spatial and

thematic evaluations, the proposed method offers a more comprehensive framework that ensures both efficiency and consistency in geospatial analysis.

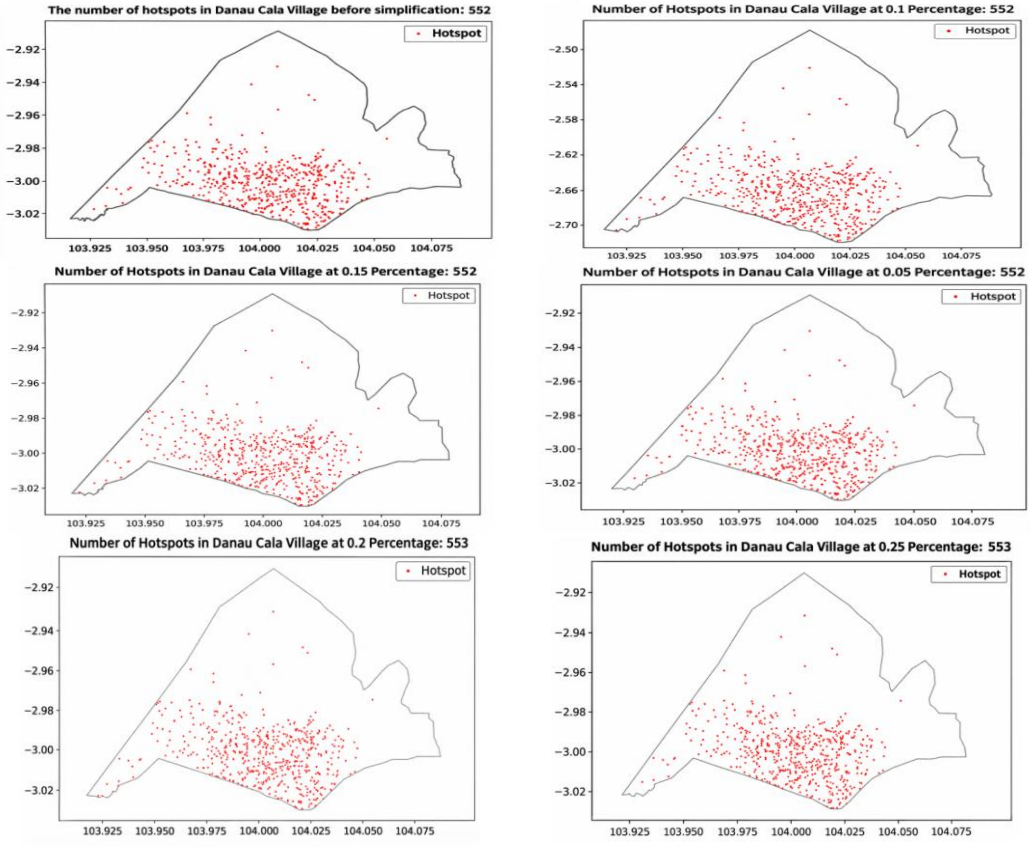


Figure 7. The number of polygon hotspots in Danau Cala Village obtained from the simplification process using percentages of 0.05, 0.1, 0.15, 0.2, and 0.25

This study evaluates adaptive metric-based thresholds for polygon simplification in South Sumatra, demonstrating a balance between geometric fidelity and thematic consistency. However, its limited geographic scope warrants further testing across provinces with diverse landscapes and polygon complexities to assess scalability and robustness.

Unlike previous studies emphasizing geometric fidelity through Hausdorff distance and shape similarity [62], this study also evaluates thematic consistency, including hotspot stability, to support fire monitoring and disaster management. The slight variation in hotspot counts observed at 0.20% and 0.25% thresholds may be attributed to boundary shifts that cause certain hotspots near village borders to be reassigned or excluded. This highlights the trade-off between aggressive simplification and thematic accuracy, reinforcing the need to select an optimal threshold that balances both.

Despite its effectiveness, this study has several limitations. First, the evaluation was limited to a single province (South Sumatra), and results may vary in regions with different geometric complexities. Second, the method assumes uniform importance of all polygons, whereas adaptive weighting based on land use or population could further improve accuracy. Future research should explore the generalizability of these findings to other regions and incorporate dynamic weighting schemes for improved adaptability.

4. CONCLUSION

A metric-based adaptive thresholding approach for the DP algorithm was developed, integrating spatial and thematic evaluations. The DP algorithm was applied to village polygons with an optimal threshold of 0.15% for calculating the NoH as an indicator of forest and land fires. The threshold value of 0.15% (based on the bounding box diagonal) provides an optimal balance between geometric fidelity and hotspot consistency. Additionally, further studies are encouraged to enhance the visualization of Forest and land fires hotspot

distribution within the simplified village polygons using an OLAP system. These results contribute to the development of an adaptive method for spatial generalization, differing from the static approaches used in previous studies, thereby supporting its application in various geospatial scenarios.

These findings provide a foundation for developing adaptive generalization techniques that integrate both geometric and thematic considerations, which has been a major gap in spatial data simplification research. For the disaster management community, this approach can enhance the accuracy of fire risk analysis while reducing computational costs, enabling real-time decision-making in systems like SiPongi or national spatial data infrastructures.

ACKNOWLEDGMENTS

We would like to thank Telkom University for funding this research. This research was conducted in collaboration with researchers at the School of Data Science, Mathematics and Informatics (SSMI), IPB University. The authors also wish to express their appreciation to IPB University for the support provided and are grateful for the assistance given by SSMI during the implementation of this research.

FUNDING INFORMATION

This research is funded by Telkom University (No: 138/SDM5/BSDM/2023).

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Rahmadi Wijaya	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓
Imas Sukaesih Sitanggang	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	
Irman Hermadi				✓	✓	✓		✓		✓		✓	✓	
Annisa					✓	✓		✓		✓		✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors state that there is no conflict of interest.

DATA AVAILABILITY

The research data used in this study are publicly available:

- Data related to Forest and land fires are available at https://firms.modaps.eosdis.nasa.gov/active_fire/. This dataset includes geolocation information (latitude and longitude), acquisition time, satellite sources (MODIS and VIIRS), and the confidence level for each detected fire point. Environmental parameters were measured at one location in South Sumatra Province, Indonesia. The measurements were conducted from January 1, 2013 to December 31, 2023.
- Data related to all villages in South Sumatra Province, Indonesia are available at https://tanahair.indonesia.go.id/portal-web. The dataset includes base maps (administrative boundaries, land cover, hydrography, hypsography, transportation networks, and toponymy) as well as thematic maps provided by various government institutions.
- Data related to Statistical Tables and Official Publications of national and regional indicators are available at https://www.bps.go.id. The dataset includes demographic statistics, national accounts, labor force, education, health, agriculture, and other socio-economic indicators.

REFERENCES

[1] M. Yazid, D. Adriani, Riswani, and D. Damayanthy, "Farm household vulnerability due to land and forest fire in peatland areas in

South Sumatra," *Land*, vol. 13, no. 5, May 2024, doi: 10.3390/land13050642.
[2] L. Hein *et al.*, "The health impacts of Indonesian peatland fires," *Environmental Health*, vol. 21, no. 1, Dec. 2022, doi: 10.1186/s12940-022-00872-w.
[3] R. Kumalawati, A. Yuliarti, M. Septiana, Syaifuddin, K. H. Murliawan, and R. N. Anggraeni, "Factors affecting forest and land fires in the media frame for future disaster mitigation," *Jurnal Ilmu Kehutanan*, vol. 16, no. 2, pp. 142–151, 2022, doi: 10.22146/jik.v16i2.2153.
[4] C. B. Engel, S. D. Jones, and K. J. Reinke, "Real-time detection of daytime and night-time fire hotspots from geostationary satellites," *Remote Sensing*, vol. 13, no. 9, Apr. 2021, doi: 10.3390/rs13091627.
[5] B. Ouattara *et al.*, "Enhancing burned area monitoring with VIIRS dataset: A case study in Sub-Saharan Africa," *Science of Remote Sensing*, vol. 10, Dec. 2024, doi: 10.1016/j.srs.2024.100165.
[6] Statistics Indonesia (BPS), "Statistics Indonesia – official statistical data portal," [Online]. Available: https://www.bps.go.id. (Accessed: Jun. 01, 2025).
[7] M.-W. Son, C.-G. Kim, and B.-S. Kim, "Development of an algorithm for assessing the scope of large forest fire using VIIRS-based data and machine learning," *Remote Sensing*, vol. 16, no. 14, Jul. 2024, doi: 10.3390/rs16142667.
[8] H. Dastour, H. Bhuian, M. R. Ahmed, and Q. K. Hassan, "Active fire clustering and spatiotemporal dynamic models for forest fire management," *Fire*, vol. 7, no. 10, Oct. 2024, doi: 10.3390/fire7100355.
[9] W. Li, E. Dodwell, and D. Cook, "A clustering algorithm to organize satellite hotspot data for the purpose of tracking bushfires remotely," *The R Journal*, vol. 15, no. 1, pp. 17–33, Aug. 2023, doi: 10.32614/RJ-2023-022.
[10] N. Maeda and H. Tonooka, "Early stage forest fire detection from Himawari-8 AHI images using a modified MOD14 algorithm combined with machine learning," *Sensors*, vol. 23, no. 1, Dec. 2022, doi: 10.3390/s23010210.
[11] T. Shiraishi, R. Hirata, M. Hayashi, and T. Hirano, "Carbon dioxide emissions through land use change, fire, and oxidative peat decomposition in Borneo," *Scientific Reports*, vol. 13, no. 1, Aug. 2023, doi: 10.1038/s41598-023-40333-z.
[12] S. A. Suab *et al.*, "Assessing Borneo's tropical forests and plantations: a multi-sensor remote sensing and geospatial MCDA approach to environmental sustainability," *Frontiers in Forests and Global Change*, vol. 7, Jul. 2024, doi: 10.3389/ffgc.2024.1337535.
[13] C. Xiao *et al.*, "Deforestation increases vegetation vulnerability to drought across biomes," *Global Biogeochemical Cycles*, vol. 39, no. 5, May 2025, doi: 10.1029/2024GB008378.
[14] Z. Zhang, Z. Qu, S. Liu, D. Li, J. Cao, and G. Xie, "Expandable on-board real-time edge computing architecture for Luojia3 intelligent remote sensing satellite," *Remote Sensing*, vol. 14, no. 15, Jul. 2022, doi: 10.3390/rs14153596.
[15] M. Tong, S. Li, X. Wang, and P. Wei, "Inter-satellite cooperative offloading decision and resource allocation in mobile edge computing-enabled satellite–terrestrial networks," *Sensors*, vol. 23, no. 2, Jan. 2023, doi: 10.3390/s23020668.
[16] K. Thangavel *et al.*, "Autonomous satellite wildfire detection using hyperspectral imagery and neural networks: A case study on Australian wildfire," *Remote Sensing*, vol. 15, no. 3, Jan. 2023, doi: 10.3390/rs15030720.
[17] M. Wijayatunga and X. Wu, "An optimised satellite constellation for forest fire detection through edge computing," *arXiv Preprint*, Sep. 2022, doi: 10.48550/arXiv.2209.07038.
[18] L. Zhang, Y. Zhou, L. Tang, S. Jia, and Z. Dai, "A self-adaptive compression method for ship trajectories without threshold setting," *Journal of Marine Science and Engineering*, vol. 12, no. 6, Jun. 2024, doi: 10.3390/jmse12060980.
[19] S. Storandt and J. Zink, "Polyline simplification under the local frechet distance has almost-quadratic runtime in 2D," *arXiv Preprint*, Jan. 2022, doi: 10.48550/arXiv.2201.01344.
[20] Y. Wen and W. Zhang, "A minimax model for generalized penetration distance between convex sets by directed Hausdorff distance," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6123–6130, Jul. 2022, doi: 10.1109/LRA.2022.3166111.
[21] I. M. Al Jawarneh, L. Foschini, and P. Bellavista, "Polygon simplification for the efficient approximate analytics of georeferenced big data," *Sensors*, vol. 23, no. 19, Sep. 2023, doi: 10.3390/s23198178.
[22] R. A. Alshamsi, I. M. Al Jawarneh, L. Foschini, and A. Corradi, "ApproxGeoMap: An efficient system for generating approximate geo-maps from big geospatial data with quality of service guarantees," *Computers*, vol. 14, no. 2, Jan. 2025, doi: 10.3390/computers14020035.
[23] B. Liu, X. Liu, D. Li, Y. Shi, G. Fernandez, and Y. Wang, "A vector line simplification algorithm based on the Douglas–Peucker algorithm, monotonic chains and dichotomy," *ISPRS International Journal of Geo-Information*, vol. 9, no. 4, Apr. 2020, doi: 10.3390/ijgi9040251.
[24] Z. Zhou, Y. Zhang, X. Yuan, and H. Wang, "Compressing AIS trajectory data based on the multi-objective peak Douglas–Peucker algorithm," *IEEE Access*, vol. 11, pp. 6802–6821, 2023, doi: 10.1109/ACCESS.2023.3234121.
[25] D. Ma, Z. Zhao, Y. Zheng, R. Guo, and W. Zhu, "PolySimp: A tool for polygon simplification based on the underlying scaling hierarchy," *ISPRS International Journal of Geo-Information*, vol. 9, no. 10, Oct. 2020, doi: 10.3390/ijgi9100594.
[26] L. Tang, L. Zhang, J. Dong, H. Wei, and S. Wei, "An adaptive simplification method for coastlines using a skeleton line 'bridge' double direction buffering algorithm," *ISPRS International Journal of Geo-Information*, vol. 13, no. 5, May 2024, doi: 10.3390/ijgi13050155.
[27] BIG 2020, "Geospatial information agency," *BIG 2020*, 2020. [Online]. Available: https://tanahair.indonesia.go.id/portal-web. (Accessed: Jun. 15, 2023).
[28] MODIS Fire Data, "MODIS fire data archive (1 January 2013 - 31 December 2023)," *MODIS Fire Data*. [Online]. Available: https://firms.modaps.eosdis.nasa.gov/active_fire/. (Accessed: Jun. 01, 2025).
[29] GeoPandas, "GeoPandas Documentation," 2025. [Online]. Available: https://geopandas.org/. (Accessed: Sep. 16, 2025).
[30] K. T. Jacobs and S. W. Mitchell, "OpenStreetMap quality assessment using unsupervised machine learning methods," *Transactions in GIS*, vol. 24, no. 5, pp. 1280–1298, Oct. 2020, doi: 10.1111/tgis.12680.
[31] K. Jetlund, *Harmonizing and linking conceptual models of geospatial information: Technologies for information modelling in GIS, ITS and BIM*. ntnuopen.ntnu.no, 2021. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2740648.
[32] F. Noardo, "Multisource spatial data integration for use cases applications," *Transactions in GIS*, vol. 26, no. 7, pp. 2874–2913, Nov. 2022, doi: 10.1111/tgis.12987.
[33] D. Zhang *et al.*, "Real-time wildfire detection algorithm based on VIIRS fire product and Himawari-8 data," *Remote Sensing*, vol. 15, no. 6, Mar. 2023, doi: 10.3390/rs15061541.
[34] K. Coskuner, "Assessing the performance of MODIS and VIIRS active fire products in the monitoring of wildfires: a case study in Turkey," *iForest - Biogeosciences and Forestry*, vol. 15, no. 2, pp. 85–94, Apr. 2022, doi: 10.3832/ifor3754-015.
[35] J. A. Moreno-Ruiz, J. R. García-Lázaro, M. Arbelo, and M. Cantón-Garbín, "MODIS sensor capability to burned area mapping—assessment of performance and improvements provided by the latest standard products in boreal regions," *Sensors*, vol. 20, no. 18, Sep. 2020, doi: 10.3390/s20185423.

[36] R. Song *et al.*, "Spatial and temporal variation of air pollutant emissions from forest fires in China," *Atmospheric Environment*, vol. 281, Jul. 2022, doi: 10.1016/j.atmosenv.2022.119156.
[37] N. C. Santana, O. A. de Carvalho, R. A. T. Gomes, and R. F. Guimarães, "Accuracy and spatiotemporal distribution of fire in the Brazilian biomes from the MODIS burned-area products," *International Journal of Wildland Fire*, vol. 29, no. 10, pp. 907–918, Oct. 2020, doi: 10.1071/WF19044.
[38] M. Breunig *et al.*, "Geospatial Data Management Research: Progress and Future Directions," *ISPRS International Journal of Geo-Information*, vol. 9, no. 2, Feb. 2020, doi: 10.3390/ijgi9020095.
[39] M. L. Campagnolo, R. Libonati, J. A. Rodrigues, and J. M. C. Pereira, "A comprehensive characterization of MODIS daily burned area mapping accuracy across fire sizes in tropical savannas," *Remote Sensing of Environment*, vol. 252, Jan. 2021, doi: 10.1016/j.rse.2020.112115.
[40] T. Zhang, Z. Wang, and P. Wang, "A method for compressing AIS trajectory based on the adaptive core threshold difference Douglas–Peucker algorithm," *Scientific Reports*, vol. 14, no. 1, Sep. 2024, doi: 10.1038/s41598-024-71779-4.
[41] M. Ma, Y. Wu, X. Ouyang, L. Chen, J. Li, and N. Jing, "HiVision: Rapid visualization of large-scale spatial vector data," *Computers & Geosciences*, vol. 147, Feb. 2021, doi: 10.1016/j.cageo.2020.104665.
[42] G. Barequet and S. Har-Peled, "Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions," *Journal of Algorithms*, vol. 38, no. 1, pp. 91–109, Jan. 2001, doi: 10.1006/jagm.2000.1127.
[43] F. Hassanzadeh and D. Rappaport, "Approximation algorithms for finding a minimum perimeter polygon intersecting a set of line segments," in *Algorithms and Data Structures (WADS 2009)*, 2009, pp. 363–374, doi: 10.1007/978-3-642-03367-4_32.
[44] Y. Choi, S. Lee, and H.-K. Ahn, "Maximum-area and maximum-perimeter rectangles in polygons," *Computational Geometry*, vol. 94, Mar. 2021, doi: 10.1016/j.comgeo.2020.101710.
[45] H. Gong, M. Zhang, J. Wang, and L. Meng, "A generic method to organize boundaries of complex polygons for embedded devices," *Arabian Journal of Geosciences*, vol. 11, no. 10, May 2018, doi: 10.1007/s12517-018-3575-6.
[46] M. Wu *et al.*, "An Efficient Visualization Method for Polygonal Data with Dynamic Simplification," *ISPRS International Journal of Geo-Information*, vol. 7, no. 4, Apr. 2018, doi: 10.3390/ijgi7040138.
[47] Y. Shen, T. Ai, and Y. He, "A new approach to line simplification based on image processing: A case study of water area boundaries," *ISPRS International Journal of Geo-Information*, vol. 7, no. 2, Jan. 2018, doi: 10.3390/ijgi7020041.
[48] Z. Li, J. Zhai, and F. Wu, "Shape similarity assessment method for coastline generalization," *ISPRS International Journal of Geo-Information*, vol. 7, no. 7, Jul. 2018, doi: 10.3390/ijgi7070283.
[49] C. Lenz, "Curvature based surface mesh simplification," Thesis, Technische Universitat Wien (Vienna University of Technology), 2019.
[50] Y. Xu, Z. Xie, Z. Chen, and M. Xie, "Measuring the similarity between multipolygons using convex hulls and position graphs," *International Journal of Geographical Information Science*, vol. 35, no. 5, pp. 847–868, May 2021, doi: 10.1080/13658816.2020.1800016.
[51] D. Sidlauskas, S. Chester, E. T. Zacharatou, and A. Ailamaki, "Improving Spatial Data Processing by Clipping Minimum Bounding Boxes," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, Apr. 2018, pp. 425–436, doi: 10.1109/ICDE.2018.00046.
[52] J. M. Jayoma, G. P. Navales, F. A. M. Bustamante, and L. R. Tradio, "Enhanced ramer-Douglas-Peucker algorithm for efficient 2D line simplification," in *2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, Nov. 2023, pp. 1–6, doi: 10.1109/HNICEM60674.2023.10589058.
[53] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica*, vol. 10, no. 2, pp. 112–122, Dec. 1973, doi: 10.3138/FM57-6770-U75U-7727.
[54] R. Wang, M. Shafeeque, H. Yan, and L. Xiaoming, "Minimizing uncertainty in complete automation of Douglas-Peucker algorithm for geospatial mapping." Dec. 29, 2021, doi: 10.21203/rs.3.rs-1162837/v1.
[55] F. A. Alhammadi, H. A. Alsuwaidi, S. A. Alzarooni, and I. M. Al Jawarneh, "Line simplification for efficient approximate join queries on big geospatial data," in *2024 Fifth International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, Sep. 2024, pp. 89–94, doi: 10.1109/IDSTA62194.2024.10747008.
[56] A. R. Ginanjar, I. S. Sitanggang, and A. Annisa, "Optimization of spatial visualization module in solap for Indonesian agricultural commodities," *International Journal of Geoinformatics*, vol. 16, no. 1, pp. 9–19, 2020.
[57] R. van Hunnik, "Extensive comparison of trajectory simplification algorithms," Thesis, Utrecht University, 2017.
[58] T. Samsonov and O. Yakimova, "Regression modeling of reduction in spatial accuracy and detail for multiple geometric line simplification procedures," *International Journal of Cartography*, vol. 6, no. 1, pp. 47–70, Jan. 2020, doi: 10.1080/23729333.2019.1615745.
[59] M. Z. Abdillah, D. A. Nawangnugraeni, S. Solikhin, T. Wijanarko, and A. Putra, "JSON and MYSQL databases for spatial visualization of polygon and multipolygon data in geographic information systems: A comparative study," *Scientific Journal of Informatics*, vol. 10, no. 4, pp. 435–444, Nov. 2023, doi: 10.15294/sji.v10i4.47393.
[60] S. Katragadda, J. Chen, and S. Abbady, "Spatial hotspot detection using polygon propagation," *International Journal of Digital Earth*, vol. 12, no. 7, pp. 825–842, Jul. 2019, doi: 10.1080/17538947.2018.1485754.
[61] L. Hatcher, "The hot spots conjecture for some non-convex polygons," *arXiv preprint*, May 2025, doi: 10.48550/arXiv.2405.19508.
[62] X. Barillot, J.-F. Hangouët, and H. Kadri-Dahmani, "Generalization of the 'Douglas and peucker' algorithm for cartographic applications," in *Proceedings of the 20th International Conference*, Beijing, China, 2001, pp. 1–8.
[63] A. A. Taha and A. Hanbury, "An efficient algorithm for calculating the exact Hausdorff distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 11, pp. 2153–2163, Nov. 2015, doi: 10.1109/TPAMI.2015.2408351.
[64] A. Javaheri, C. Brites, F. Pereira, and J. Ascenso, "A generalized Hausdorff distance based quality metric for point cloud geometry," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, May 2020, pp. 1–6, doi: 10.1109/QoMEX48832.2020.9123087.
[65] X. Lei, Y. Ding, and F.-X. Wu, "Detecting protein complexes from DPINs by density based clustering with pigeon-inspired optimization algorithm," *Science China Information Sciences*, vol. 59, no. 7, Jul. 2016, doi: 10.1007/s11432-016-5578-9.

BIOGRAPHIES OF AUTHORS



Rahmadi Wijaya received the bachelor's degree (B.Sc.) in Computer Science from Universitas Padjadjaran (UNPAD), Bandung, Indonesia, in 1999, and the master's degree in information systems from the Institut Teknologi Bandung (ITB) in 2006. He is currently pursuing a Ph.D. at Institut Pertanian Bogor (IPB) University, Indonesia. He is currently a Lecturer in Software Engineering at Telkom University, Bandung, Indonesia. His research interests include software engineering, geospatial data machine learning, and the integration of GIS into web-based systems. He can be contacted at email: wijayarahmadi@apps.ipb.ac.id or wijayarahmadi@apps.ipb.ac.id.



Imas Sukaesih Sitanggang is a lecturer in the School of Data Science, Mathematics and Informatics IPB University. Her research areas are focused on spatial data mining and smart agriculture. She can be contacted at email: imas.sitanggang@apps.ipb.ac.id.



Irman Hermadi is associate professor and member of the Software Engineering and Information Science Division (since 2013) at the Computer Science Study Program at IPB University. He got a B.Sc. in Computer Science from IPB University in 1999 and immediately joined Telkom University as a lecturer for 18 months. He pursued his master's study at King Fahd University of Petroleum & Minerals in computer science. His work on path testing using genetic algorithms at the University of New South Wales, Australia, has awarded him a Ph.D. degree in Computer Science in 2015. He has over 22 years of academic, research, and professional experience in computer science, computational intelligence, software engineering, and information systems. He can be contacted at email: irmanhermadi@apps.ipb.ac.id.



Annisa is a senior lecturer in the Department of Computer Science at IPB University. Her research interests include data mining, skyline query, and recommender system. Her current research focuses on data mining and application of skyline queries for recommended systems in various fields. She can be contacted at email: annisa@apps.ipb.ac.id.