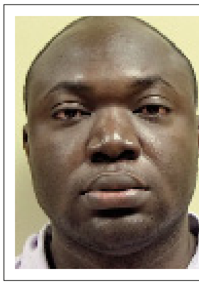
CONTEXTUAL DOUGLAS-PEUCKER SIMPLIFICATION

Titus Tienaah*, Emmanuel Stefanakis, David Coleman

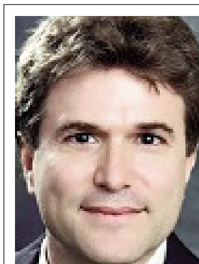
Geodesy and Geomatics Engineering
University of New Brunswick, Fredericton, New Brunswick

In this paper, we develop a constrained Douglas-Peucker algorithm using a polyline to be simplified and other geometries as contextual constraints. We develop a contextual model that incrementally rewinds to the original polyline with relevant characteristic vertices to resolve contextual conflicts. Constraints covered in this paper are topology and direction. Our implementation shows a consistent representation and a technique to accelerate multi-scale simplification of polylines.

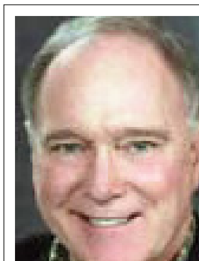
Dans cet article, nous développons un algorithme de Douglas-Peucker restreint utilisant une polyligne à simplifier et d'autres géométries comme contraintes contextuelles. Nous élaborons un modèle contextuel qui retourne par incrément à la polyligne originale à l'aide des sommets caractéristiques pertinents pour résoudre les conflits contextuels. La topologie et la direction constituent les contraintes abordées dans cet article. Notre mise en œuvre démontre une représentation uniforme et présente une technique pour accélérer la simplification multi-échelle des polylignes.



Titus Tienaah
ttienaah@unb.ca
**Corresponding author*



Emmanuel Stefanakis
estef@unb.ca



David Coleman
dcoleman}@unb.ca

1. Introduction

Line simplification is a fundamental process in cartographic generalization [*Cromley* 1991; *Weibel* 1997]. As an operator, simplification reduces redundancy or the level of detail required to represent a polyline at a desired scale. Line simplification is a well-covered topic with many algorithms in cartography, computational geometry and computer graphics, e.g., *Ramer* [1972], *Douglas and Peucker* [1973], *Reumann and Witkam* [1974], *Opheim* [1982], *Li and Openshaw* [1992], *Normant and Tricot* [1993], V*isvalingam and Whyatt* [1993], *Fritsch and Lagrange* [1995], *Balboa and López* [2000], *Doihara et al.* [2002], *Qingsheng et al.* [2002], *Wu and Deng* [2003], *Saux* [2003], *Gribov and Bodansky* [2004], *Kulik et al.* [2005], *Guilbert and Saux* [2008], *Abam et al.* [2010], *Daneshpajouh and Ghodsi* [2011], *Lei et al.* [2012], *Liu et al.* [2012], *Raposo* [2013] and *Abam et al.* [2014]. One of the most popular algorithms in line simplification is the Douglas-Peucker (DP) algorithm [*Douglas and Peucker* 1973]. The DP algorithm, like many other techniques of simplification, has an implicit scope to reduce the set of vertices used to represent a polyline based on some criteria (distance offset, error bound, direction/angular deflection, area and other measures). The reduction technique is simple, but obtaining a consistent or meaningful output is a complicated problem. The issues of consistent simplification arise from the fact that real world geometric properties are often constrained by neighbouring objects within a given space. Simplification of geometric properties without context may change the meaning of such properties (semantic representation).

In recent times, there is a greater need to perform simplification due to multi-scale representation of online interactive maps, as well as high spatial and temporal resolution of data collected using location-enabled devices equipped with Global Positioning System (GPS). Evidence of the need for line simplification is observable in widely used Web and desktop mapping libraries or applications [*ESRI ArcGIS* 2015; *Leaflet* 2015; *Openlayers* 2015; *PostGIS* 2015, *Oracle* 2015]. As part of the Leaflet Geometry module, L.LineUtil (*http://bit.ly/1zdZ3nD*), provides utility functions for line simplification using the DP algorithm to improve vector map rendering. OpenLayers geometry library, ol.geom.-flat.simplify.douglasPeucker (*http://bit.ly/1vLJvaq*), is based on the library extracted from Leaflet. PostGIS, a spatial database extension, uses ST_SimplifyPreserveTopology (*http://bit.ly/1vhwBCg*) to perform topology preserving simplification. ESRI provides SimplifyLine_cartography as part of their cartographic toolset in ArcGIS10.2.2. The PointRemove option of Simplify-Line_cartography uses DP line simplification. Oracle Spatial implements the DP algorithm as sdo_util.simplify (*http://bit.ly/1JTdbv3*).

The main contribution of this paper is a contextual model of the Douglas-Peucker algorithm with a prioritized context-based reversion to maintain topology and direction relation in an embedded space. We achieve consistent simplification by: (1) developing a line characteristic and constrained simplification model (section 2.1); (2) extending a directional relation operator/signature (section 2.2); and (3) extending the Binary Line Generalization Tree (section 3).

2. Conceptual Definitions

A polyline (e.g., road centre line or coastline) is composed of points as vertices (e.g., location or intersection) and can be composed to form polygons (e.g., administrative regions, soil, forest and other thematic features). We use a modified subset of the OGC (Open Geospatial Consortium) simple feature specification of a LineString, Line and LinearRing [*OGC* 2014]. This paper defines a polyline (or often referred to as a line) as a continuous chain of segments. As shown in Figure 1, a segment consists of a simple straight line connecting two points. A segment has a length property if the ends of the segment are separate in two-dimensional Euclidean space. A segment with zero length (coincident end points) can be generalized as a point. A polyline is simple if it does not self-intersect, and is a ring if the end point of the last segment is the beginning point of the first segment. A self-intersecting ring forms a complex ring.

Simplification of a line involves removal of redundant vertices based on a given criteria to reduce complexity in a dataset. Very often, linear simplification algorithms start with a polyline L made up of two endpoints and an arbitrary set of vertices V. With a given criteria, L is simplified into a polyline L' by reducing the number of vertices of V to V', while keeping the ends of the polyline fixed. After simplification, V' is either a proper subset of V or equivalent to V; no new vertices are introduced or displaced [*Douglas and Peucker* 1973; *Weibel* 1997]. The classical criteria that guide vertex elimination are: minimize line distortion (no vertex of L should be further away from L' than a maximum error threshold, ε_T); minimize V' (increase number of removed vertices or reduce data size); and minimize computational complexity (reduce cost of handling/rendering massive data).

In this paper, we perform constrained simplification of a polyline using the Douglas-Peucker algorithm. The simplification follows the constraint that the geometric properties of a line and its relation to other neighbouring objects should be preserved [*Mark* 1989; *Stefanakis* 2012]. Given an ordered set of N vertices ($V_{1...N}$) forming a polyline L, the Douglas-Peucker algorithm starts by marking the end points as "keep" (V'_1 *and* V'_N). The algorithm finds the vertex V_K from V_2 to V_{N-1} with the maximum error offset (ε_K) from the line joining the end points (V'_1 *and* V'_N). If the ε_K at V_K is greater or equal to a pre-selected threshold ε_T (where $\varepsilon_T > 0$), the vertex is marked as "keep" (V'_K). This process is repeated recursively by splitting the polyline at V_K as two sub-polylines $L_1(V_{1...K})$, and $L_2(V_{k...N})$ [*Douglas and Peucker* 1973]. Figure 2 shows a graphical illustration of the algorithm. The recursion terminates if the maximum error offset (ε_K) is less than ε_T or the polyline reduces to a line with only two vertices. The generalized line consists of all vertices marked as

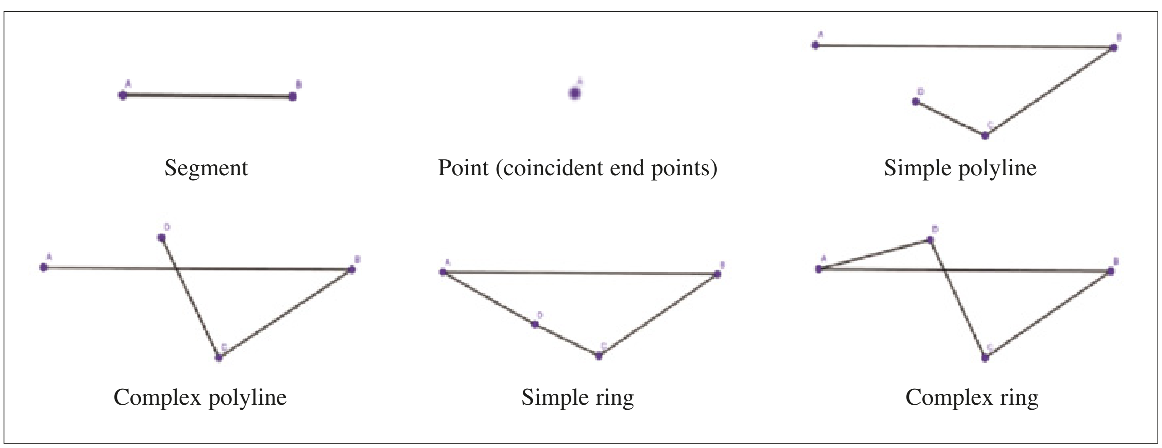
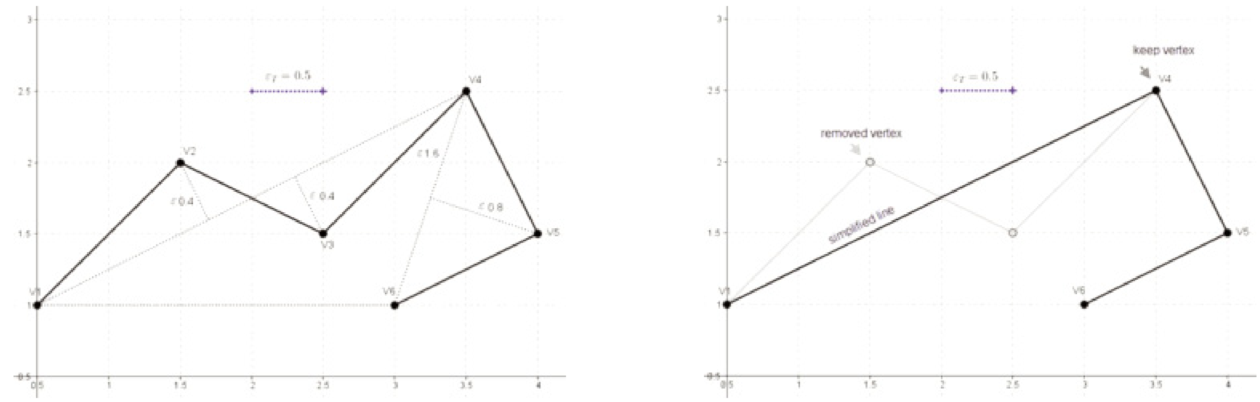


Figure 1: Line abstraction types.



(a) Original polyline. (b) Simplified polyline.

Figure 2: Douglas-Peucker simplification at 0.5 units maximum error threshold ($\varepsilon_T = 0.5$).

"keep." Worst-time complexity for the Douglas-Peucker algorithm is $0(N^2)$ and has been improved to $0(N \log N)$ [*Hershberger and Snoeyink* 1992].

A polyline on a map may exist with other spatial or thematic features. Neighbourhood geometries are context geometries that give some spatial or thematic meaning to a polyline on a map. We define neighbourhood geometry as geometries intersecting the convex hull of vertices forming the polyline. The convex hull forms a polygon with a minimum set of vertices that envelop all the vertices of the polyline. A simplified line is a subset of all vertices of the original polyline; such a line is completely within or shares boundary with the convex hull. Other geometries that intersect the convex hull may have disjoint, intersect or side relationship with the original polyline. A neighbouring geometry in relation to a polyline to be simplified can be a segment, polyline, point or polygon. For example, Figure 3 illustrates the concept of neighbours with respect to convex hull (A).

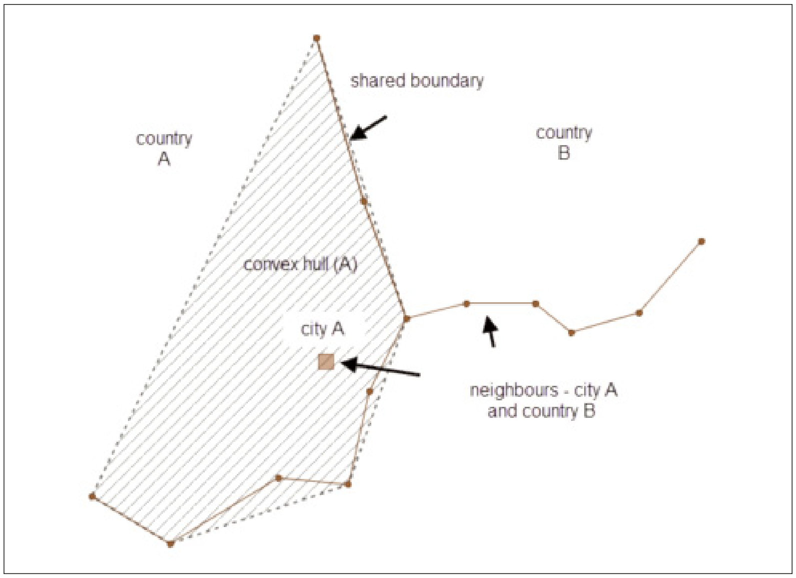
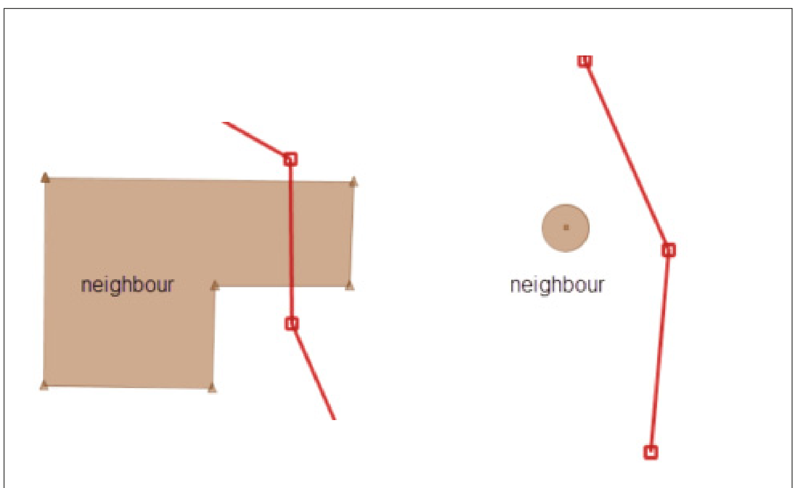


Figure 3: Neighbour geometries.

2.1 Topological Relation

Polylines have internal (simple or self-intersection) or external topological relationships with other neighbouring geometries (intersect, disjoint, left, right, top or down) (Figure 4). During Douglas-Peucker simplification, the original polyline deforms into a new geometry with a smaller set of vertices; topological relations with other geometries may be violated if these geometries are found within the convex hull of the set of vertices forming the line [*Saalfeld* 1999; *Bertolotto and Zhou* 2007; *Daneshpajouh and Ghodsi* 2011; *Stefanakis* 2012]. Our contextual model follows a

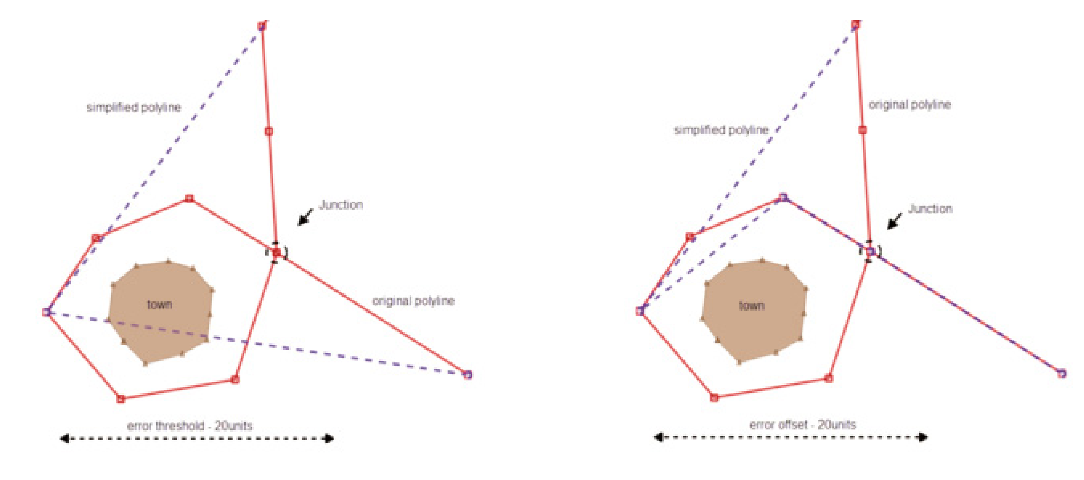


(a) Intersection. (b) Disjoint.

Figure 4: Intersect and disjoint geometric relation.

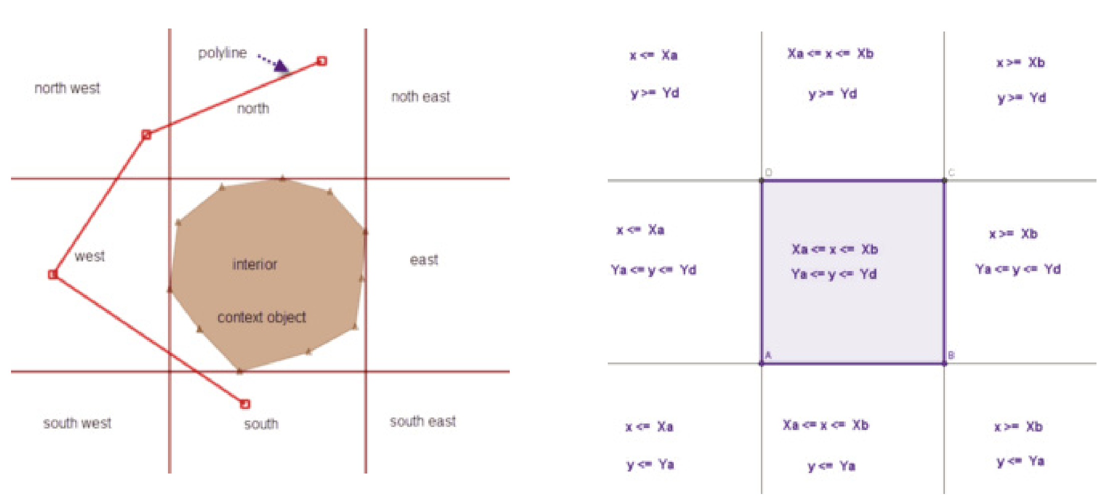
strict geometric topological rule: the topological signature of the original and simplified polyline should be the same. Geometric relations are computed using the Dimensionally Extended Nine-Intersection Model (DE-9IM) [*Egenhofer and Franzosa* 1991; *Clementini et al.* 1993]. For example, a simplified road going through a park (a region or a polygon) is valid if the original geometry shares the same relation; else, it is an inconsistent representation.

Various attempts have been made to avoid self-intersection introduced as a result of DP simplification [*Saalfeld* 1999; *Mantler and Snoeyink* 2000; *Estkowski and Mitchell* 2001; *Bertolotto and Zhou* 2007; *Pallero* 2013; *Shi and Charlton* 2013]. Real-world linear geometries sometimes self-intersect. It is important to maintain self-intersection of complex polylines such as network polylines [*Lee* 2004; *Warith* 2008] using self-intersections as constraints. The focus for most research publications regarding the DP algorithm has been attempts to avoid self-intersection for simple polylines. A typical example is a road geometry that goes around a town, roundabout or an obstacle, and self-intersects at a junction. Figure 5 (5a and 5b) demonstrates DP simplification using Java Topology Suite, version 1.8.0 [*Vivid-Solutions* 2014]. JTS is a library for 2D spatial predicate functions and



(a) JTS DP simplification. (b) JTS topology preserving DP simplification.

Figure 5: A complex polyline with self-intersection. Original self-intersection is not preserved (junction).



(a) Nine-cell side relation model. (b) Relation of polyline ([x, y]…) and MBR (ABCD).

Figure 6: Direction relation model.

spatial operators; it is one of the core and widely-used libraries in open source GIS (C++ port is used in PostGIS, GDAL/OGR, MapServer, QGIS, Shapely-Python).

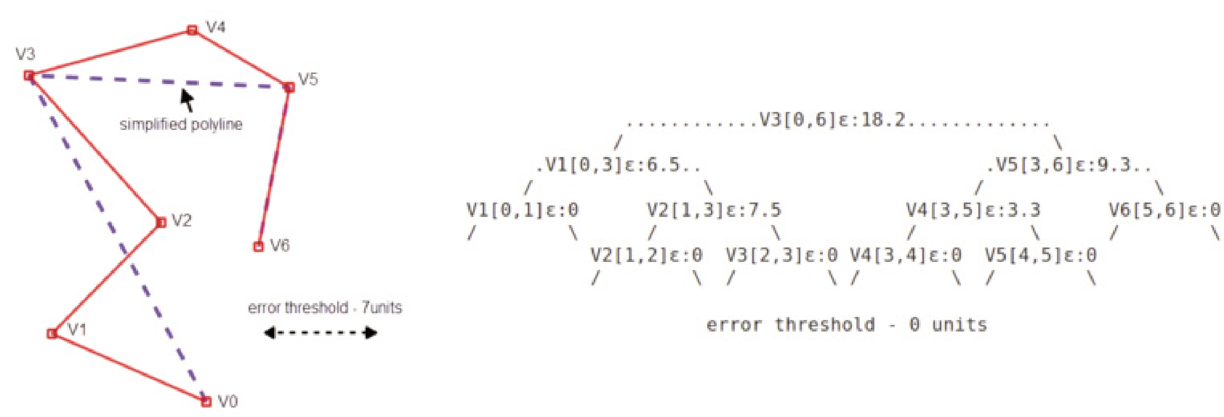
2.2 Direction Relation

The direction relationship of a polyline describes its side relation with a neighbouring object. The simplified geometry must conform to the same quadrant relation as the original to avoid side conflicts (left, right, up, down, diagonal and interior). We extend the work of *Theodoridis et al.* [1998]. Figure 6a illustrates a nine-cell direction relation model. It describes a quadrant-based relationship between a line to be simplified and the object within its planar space. By registering the quadrants in which a line intersects relative to a neighbour, we create a directional signature for which the simplified version of the line must have to be semantically consistent. Figure 6b illustrates the quadrant relationship of a polyline (with coordinates [*x, y*] …) in relation to the minimum bounding box (MBR) ABCD (with coordinates *A*[*Xa,Ya*], *B*[*Xb,Yb*], …) of a neighbouring object. We trace out the presence of a polyline in each quadrant. The intersection with a quadrant is represented as "T" for true and "F" for false. The direction relation is therefore an ordered set of T or F starting from: north-west, north, north-east, west, interior, east, south-west, south and south-east. In Figure 6a, the linear geometry has a north, north-west, west, interior and south relation with respect to the polygon (neighbour) that intersects the convex hull of its vertices. The direction relation in Figure 6a is therefore TTFTTFFTF. In Figure 5b, the topology preserving simplification does not take into account the context object and hence violates a southward direction relation.

3. Contextual Rewind Model

The idea of building hierarchical structures from polylines started in the early 1980s. *Ballard* [1981] published a hierarchical representation using a binary tree called the "Strip Tree." The strip tree structure is a direct consequence of using a special method for digitizing lines and retaining all intermediate steps, each strip (oriented minimum bounding rectangle) consisting of vertices of a polyline at each node. *Van Oosterom* [1991] introduced the Binary Line Generalization-tree (BLG-tree), which is a binary tree structure of the Douglas-Peucker algorithm.

As part of his research, the senior author extends the Binary Line Generalization tree (BLG-tree) [*Van Oosterom* 1991] by keeping a prioritized list of all the intermediate error offsets (ε) and a convex hull of vertices forming the sub-polyline at each stage of the polyline decomposition (DP algorithm). The extended BLG-tree is a direct binary tree structure of the DP algorithm. Each node of the tree has a reference to the two end points of the sub-polyline. The idea learned from the BLG-tree is to pre-process the polyline using an error threshold of zero (0) to obtain a binary tree structure of the original polyline (see Figure 7b). The extra storage (a prioritized list and convex hull of vertices at each node) in this extended BLG-tree



(a) DP simplification.

(b) BLG-tree—excluding V_0.

Figure 7: Extended BLG-tree.

structure is required for a restoration of relevant characteristic vertices (minimum set) given contextual information.

Keeping the end points of the original polyline fixed, a vertex with maximum error offset (ε) forms the top-level node. From Figure 7b, if the ε value of the root node (V3) is less than the predetermined simplification threshold (ε_T), the simplified set of vertices is a null set (∅). Line simplification as applied in the DP algorithm involves a search for the minimum set of vertices to represent the original line at a threshold. A DP simplification in the context of other planar objects requires taking into consideration the relationship between the line and other geometries. In instances where the relationship is violated by the simplified line, it is important to restore some of the vertices removed, hence a rewind towards the original line.

Figure 8 shows a DP contextual rewind model developed by the senior author of this paper. All geometries that may become neighbours are indexed using an R-tree. The model starts with a polyline; using ε_T as zero, we pre-process the polyline into a binary search tree. With a pre-processed binary tree structure, the simplification at a given ε_T is a binary search for farthest nodes with $\varepsilon \leq \varepsilon_T$ from the root node (depth-first search). A search path terminates if a node with $\varepsilon \leq \varepsilon_T$ is encountered. Using the convex hull at each node, we search the R*-tree for neighbours that intersect the convex hull of the polyline being simplified. Where a geometric relation or direction is violated, the next vertex from a prioritized list of offsets (ε) is added to the simplified sub-polyline at that node. Conflict resolution terminates when the topology and direction relations are resolved. In some highly constrained cases, the algorithm will "rewind"—or revert—to the original geometry (i.e., simplification cannot proceed without violating a topology or direction relation of the original polyline). See the "rewind loop" in Figure 8.

Note that the binary tree structure is not balanced and its efficiency depends on the shape of the original polyline. In the worst case, the binary tree reduces to a link list of nodes. Furthermore, nodes down the tree may have higher ε values than its parent node (see Figure 7b, ε value of node V2 vs. the parent V1) [*Poorten et al.* 2002]. The pre-processed binary tree structure accelerates finding characteristic vertices from a global decomposition as applied in the DP algorithm. The structure also allows multilevel simplification since the entire line is represented once in the binary tree.

Figures 9a and 9b, respectively, show an unconstrained and constrained DP simplification;

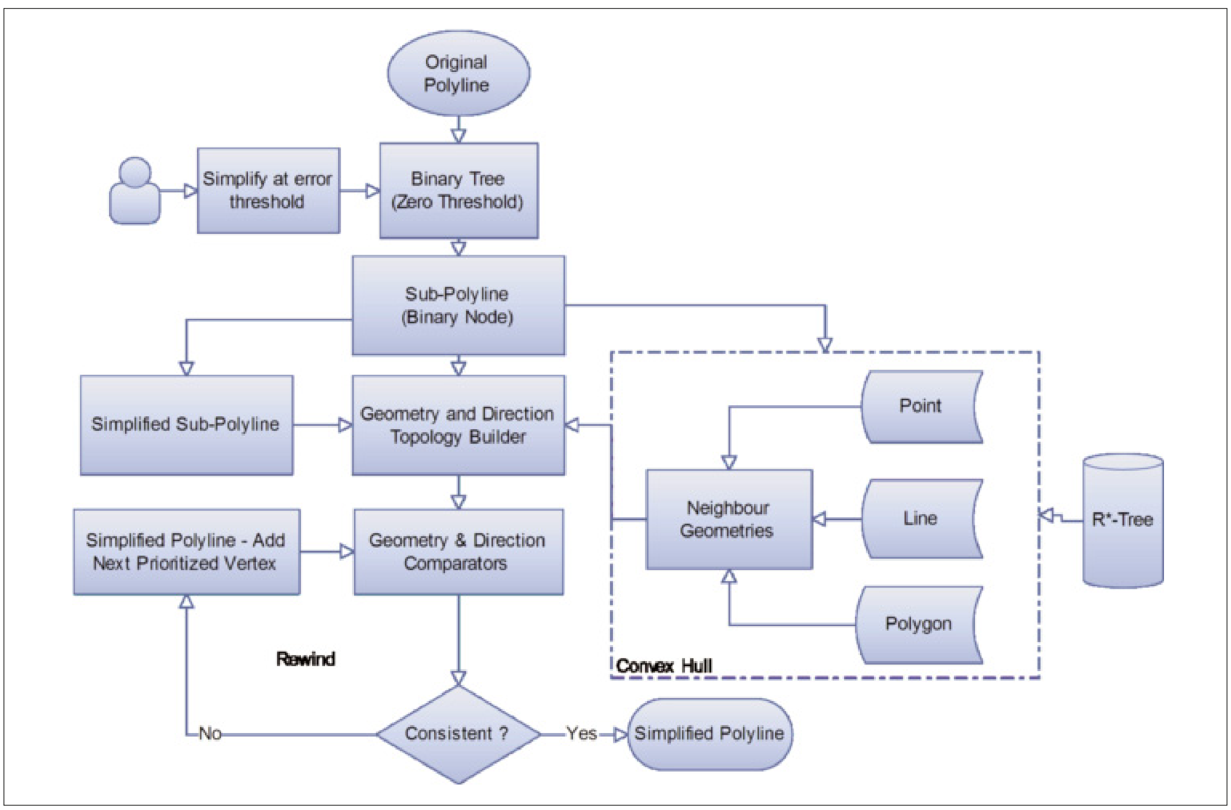
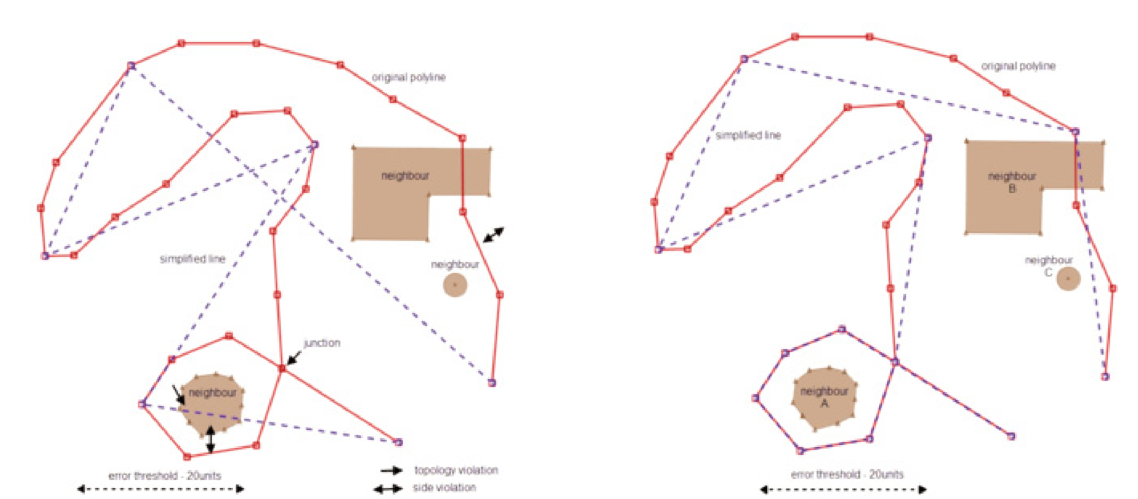


Figure 8: Contextual rewind model.

self-intersections resulting from DP simplification are not addressed in this example. Self-intersection consistency has been explored by many authors in cartography and computational geometry [*Saalfeld* 1999; *Mantler and Snoeyink* 2000; *Wu and Márquez* 2003; *Wu et al.* 2004; *Bertolotto and Zhou* 2007; *Corcoran et al.* 2011; *Li et al.* 2013]. In Figure 9a, both topology and direction relation are violated because the planar geometries as neighbours are not considered during simplification. Figure 9b illustrates a consistent (preserves topology and direction) simplified line with contextual neighbours as constraints.

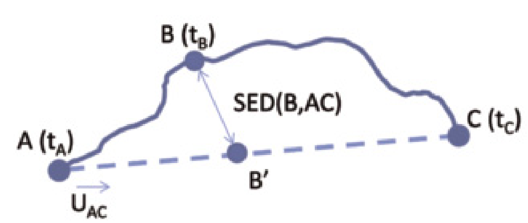
4. Trajectory Extension

The input data for the DP algorithm is an ordered set of points. A trajectory is a time-ordered set of positions of a moving object. Essentially, it is a chain of segments joining points with a temporal (time) component. The DP algorithm by itself is not able to consider the temporal dimension of a trajectory. This is achieved by introducing the notion of the Synchronous Euclidean Distance (SED) [*Meratnia and de By* 2004]. In a trajectory, each point P_i is assigned a temporal stamp (t_i), which indicates the time a moving object crossed P_i. A, B, and C are three spatiotemporal locations recorded for a trajectory T, with $t_A < t_B < t_C$ (Figure 10a). The SED for the point B is equal to the Euclidean distance BB', where the location B' is the position B on the simplified line (dash line) AC with respect to the velocity vector U_{AC} (Figure 10b). In other words, the offset (ε) of B is the distance from point B to B', where B' is the spatiotemporal trace of B on the straight-line approximation AC at time t_B. The notion of SED only changes how we compute the offset (ε) for the DP algorithm.



(a) (b)

Figure 9: Unconstrained (a) and constrained (b) DP simplification.



$$U_{AC} = \frac{AC}{t_C - t_A}$$

$$AB' = U_{AC}(t_B - t_A)$$

(a) (b)

Figure 10: The Synchronous Euclidean Distance.

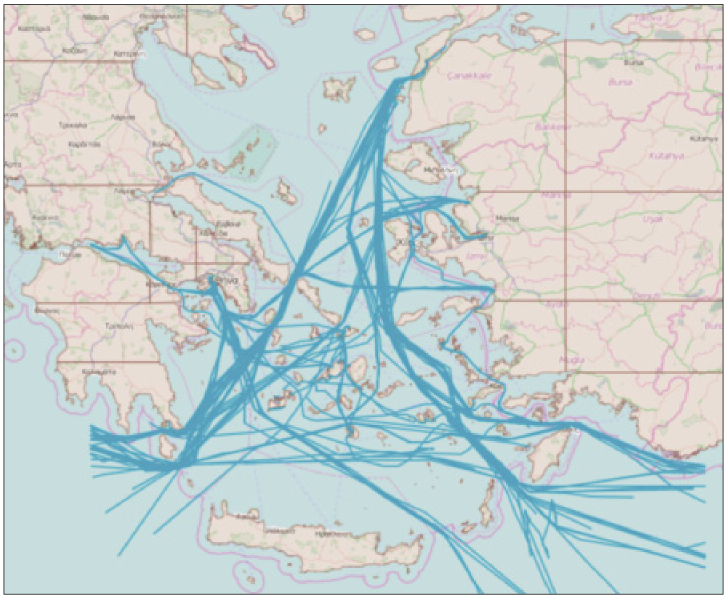


Figure 11: Study area, Aegean Sea.

5. Evaluation

The DP and SED algorithms are implemented using *Node.js*, a JavaScript platform on top of Google's V8 JavaScript engine. Vessel Trajectory Data is obtained from Marine Traffic AIS (*www.marinetraffic.com*) as comma-delimited files (CSV) with the following fields: vessel ID, latitude, longitude, time, speed, course and other attributes. The project area is the Aegean Sea, between the main lands of Greece and Turkey. This site provides a suitable set of islands that act as contextual constraints in the same planer space as the trajectories. Figure 11 shows a set of 100 trajectories in the Aegean.

A sample trajectory with 1 243 vertices is as shown in Figure 12. Each dot represents a vertex with spatiotemporal data.

Using the same trajectory illustrated in Figure 12, a constrained SED simplification at 5-km distance threshold is illustrated in Figure 13. A

Figure 12: Sample trajectory.

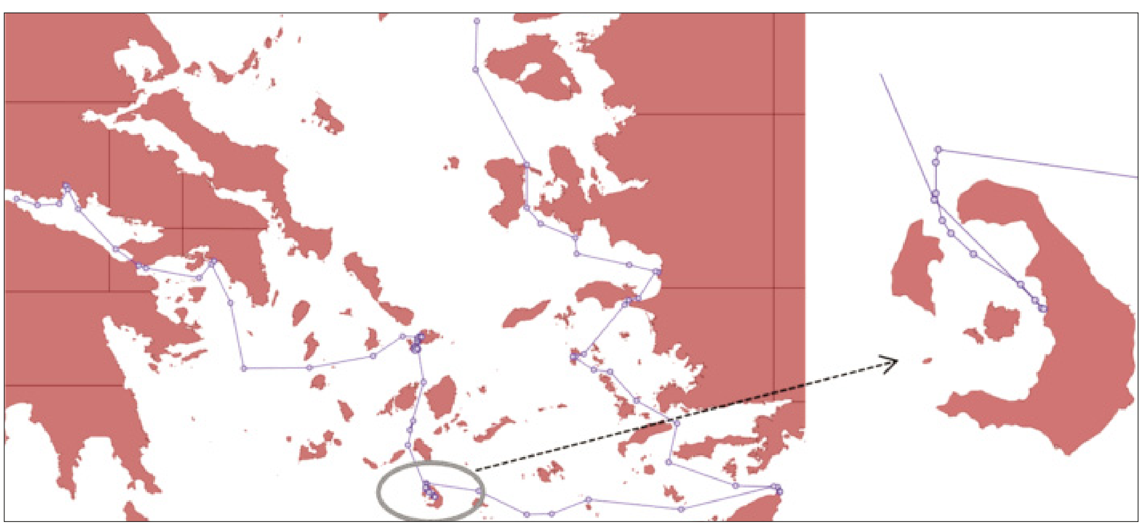


Figure 13: Constrained SED simplification at 5-km distance threshold.

comparative simplification without contextual constraints is illustrated in Figure 14. In Figure 14, the disjoint relation as shown in Figure 12 is violated. Figure 15a illustrates a section of an original trajectory and Figure 15b shows both constrained and unconstrained SED simplification. It can be observed in Figure 15b that the simplified line at 5-km threshold violates both topology and direction if not constrained.

To demonstrate the effectiveness of context-based simplification, we perform a post-empirical evaluation of constrained and unconstrained SED simplification at SED thresholds starting from 5-km to 50 km. Figure 16 illustrates percentage of vertices removed (PVR) versus SED thresholds. PVR is computed as a ratio of vertices removed as a result of simplification to the total number of vertices in the original trajectory. Figure 16 shows that compression ratio decreases with increasing offset SED distance in constrained simplification as compared to a gradual increase in unconstrained simplification.

The context-based simplification presented in this paper restores removed vertices to resolve topology and direction conflicts; as a result, the simplified line is less displaced from its original. In Figure 17, we show a total area of polygonal displacement (TAPD) [*McMaster* 1987]. TAPD is the sum of all displacement polygons standardized by the length of the original line.

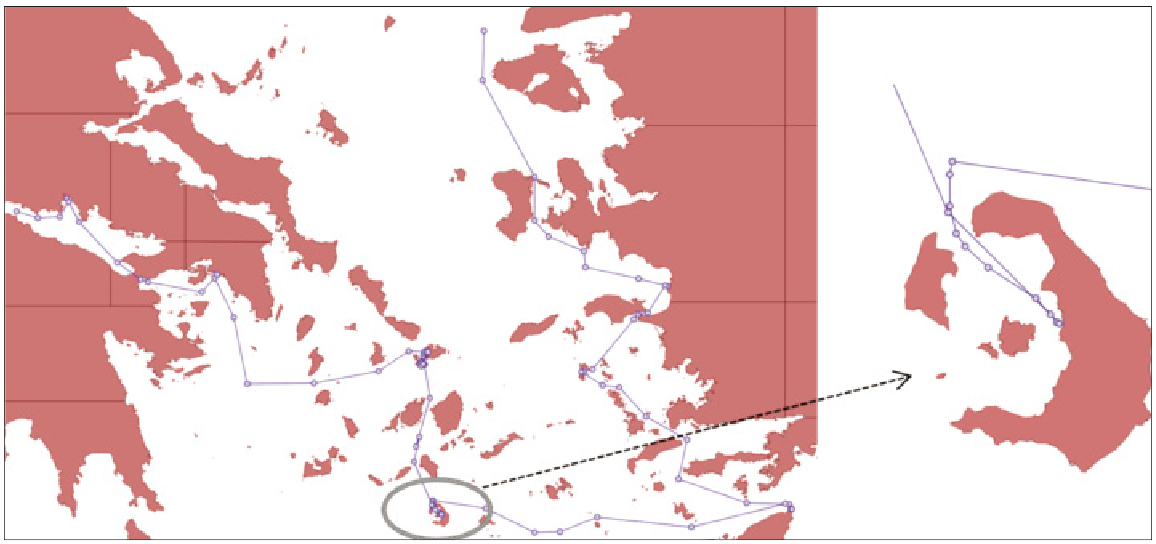
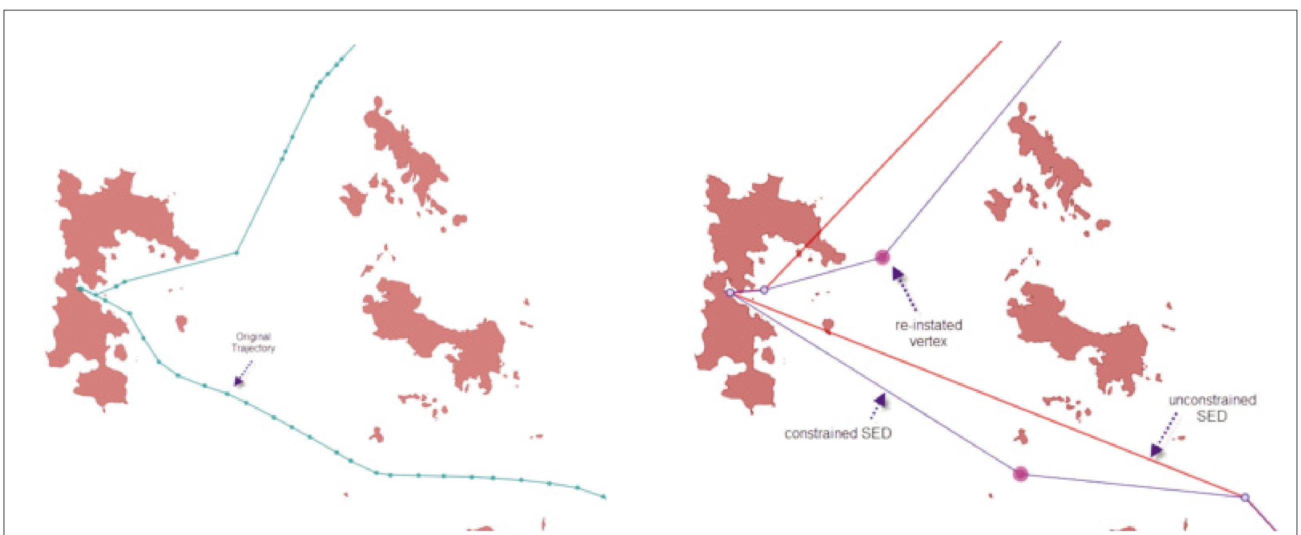


Figure 14: Unconstrained SED simplification at 5-km distance threshold.



(a) Original trajectory. (b) Constrained and unconstrained simplification.

Figure 15: Constrained and unconstrained trajectory simplification (SED).

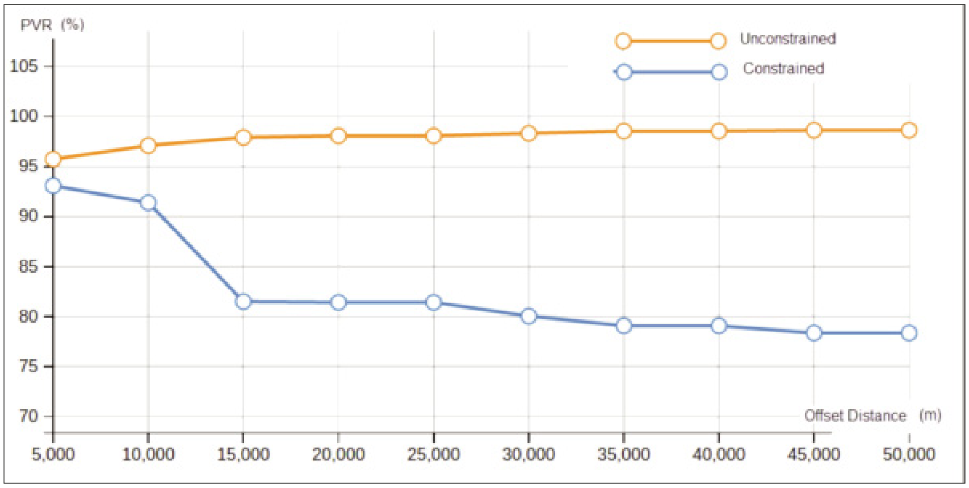


Figure 16: Percentage of redundant vertices (PVR versus ε).

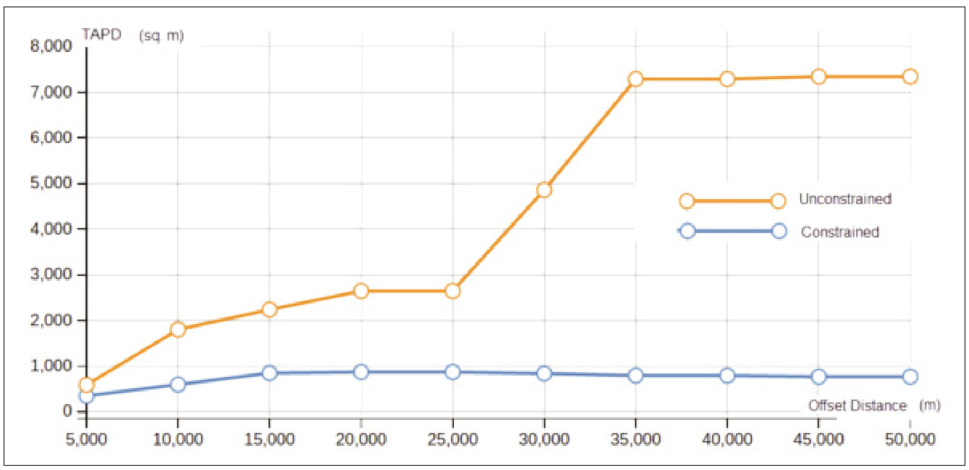


Figure 17: Total area of polygonal displacement (TAPD versus ε).

Conclusion

In this paper, we developed a contextual rewind model by extending the binary line generalization tree (BLG-tree) based on Douglas-Peucker algorithm. We incorporate a stepwise topology and direction constraint at each binary node to maintain a consistent simplification. Constraint violations are resolved by restoring a set of prioritized vertices to resolve conflicts. The binary tree structure requires storage of all intermediate steps during pre-processing (performed once at a threshold of zero). The storage penalty of the binary tree structure offsets the cost of re-computing offset distances at different offset representations of the same polyline in a multi-scale multi-representation environment. Future work will focus on localization of static and moving space constraints and other empirical evaluation metrics.

References

Abam, M.A., M. Berg, P. Hachenberger, and A. Zarei. 2010. Streaming algorithms for line simplification. *Discrete and Computational Geometry.* 43(3): 497–515.

Abam, M.A., S. Daneshpajouh, L. Deleuran, S. Ehsani, and M. Ghodsi. 2014. Computing homotopic line simplification. *Computational Geometry*. 47(7): 728–739.

Balboa, J.L.G., and F.J.A. López. 2000. Frequency filtering of linear features by means of wavelets: a method and an example. *Cartographic Journal.* 37(1): 39–49.

Ballard, D.H. 1981. Strip trees: a hierarchical representation for curves. *Commun. ACM.* 24(5): 310–321.

Bertolotto, M., and M. Zhou. 2007. Efficient and consistent line simplification for web mapping. *International Journal of Web Engineering and Technology.* 3(2): 139–156.

Clementini, E., P. Di Felice, and P. Van Oosterom. 1993. A small set of formal topological relationships suitable for end-user interaction. Ed. *Advances in Spatial Databases.* 277–295.

Corcoran, P., P. Mooney, and A. Winstanle. 2011. Planar and non-planar topologically consistent vector map simplification. *International Journal of Geographical Information Science.* 25(10): 1659–1680.

Cromley, R.G. 1991. Hierarchical methods of line simplification. *Cartography and Geographic Information Science.* 18(2): 125.

Daneshpajouh, S., and M. Ghodsi. 2011. A heuristic homotopic path simplification algorithm. In: B. Murgante, et al., Eds. *Computational Science and its Applications.* ICCSA 2011. Springer Berlin Heidelberg. 132–140.

Doihara, T., P. Wang, and W. Lu. 2002. An adaptive lattice model and its application to map simplification. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences.* 34(4): 259–262.

Douglas, D.H., and T.K. Peucker. 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(2): 112–122.

Egenhofer, M.J., and R.D. Franzosa. 1991. Point-set topological spatial relations. *International Journal of Geographical Information System.* 5(2): 161–174.

ESRI ArcGIS. 2015. Cartography toolbox: SimplifyLine_cartography (http://bit.ly/1ycbS52) [online]. Available from: http://resources.arcgis.com/en/home/ [Accessed 14 May 2015].

Estkowski, R., and J.S.B. Mitchell. 2001. Simplifying a polygonal subdivision while keeping it simple. *Proceedings of the seventeenth annual symposium on Computational geometry.* Medford, Massachusetts, USA. ACM. 40–49.

Fritsch, E., and J. Lagrange.1995. Spectral representations of linear features for generalisation. In: A. Frank, and W. Kuhn, Eds. *Spatial Information Theory A Theoretical Basis for GIS.* Springer Berlin Heidelberg. 157–171.

Gribov, A., and E. Bodansky. 2004. A new method of polyline approximation. In A. Fred, T. Caelli, R. W. Duin, A. Campilho, and D. De Ridder, Eds. *Structural, Syntactic, and Statistical Pattern Recognition.* Springer Berlin Heidelberg. 3138: 504–511.

Guilbert, E., and E. Saux. 2008. Cartographic generalisation of lines based on a B-spline snake model. *Int. J. Geogr. Inf. Sci.* 22(8): 847–870. doi: 10.1080/13658810701689846

Hershberger, J.E., and J. Snoeyink. 1992. Speeding up the Douglas-Peucker line-simplification algorithm. University of British Columbia, Department of Computer Science.

Kulik, L., M. Duckham., and M. Egenhofer. 2005. Ontology-driven map generalization. *Journal of Visual Languages and Computing.* 16(3): 245–267.

Leaflet. 2015. L.LineUtil.simplify (http://bit.ly/1zdZ3nD) [online]. Available from: https://github.com/Leaflet/Leaflet/blob/master/src/geometry/LineUtil.js [Accessed 14 May 2015].

Lee, D. 2004. Geographic and cartographic contexts in generalization. In ICA Workshop on Generalisation and Multiple Representation. Leicester, UK.

Lei, W., G. Yiming, C. Chen, and Y. Yaowei. 2012. A spatio-temporal data compression algorithm. Paper presented at the Multimedia Information Networking and Security (MINES), Fourth International Conference.

Li, L., et al. 2013. An algorithm for fast topological consistent simplification of face features. *Journal of Computational Information Systems.* 9(2): 791–803.

Li, Z., and S. Openshaw. 1992. Algorithms for automated line generalization based on a natural principle of objective generalization. *International Journal of Geographical Information Systems.* 6(5): 373–389.

Liu, G., M. Iwai, and K. Sezaki. 2012. A method for online trajectory simplification by enclosed area metric. Paper presented at the Proc. of the Sixth International Conference on Mobile Computing and Ubiquitous Networking.

Mantler, A., and J. Snoeyink. 2000. Safe sets for line simplification [online]. Available from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.402&rep=rep1&type=pdf

Mark, D.M. 1989. Conceptual basis for geographic line generalization. ed. *Proc. 9th International Symposium on Computer-Assisted Cartography.* 68–77.

McMaster, R.B. 1987. The geometric properties of numerical generalization. *Geographical Analysis.* 19 (4): 330–346.

Meratnia, N., and R. de By. 2004. Spatiotemporal compression techniques for moving point objects. In: BERTINO, E., et al., Eds. *Advances in Database Technology.* EDBT 2004. Springer Berlin Heidelberg. 765–782.

Normant, F., and C. Tricot. 1993. Fractal simplification of lines using convex hulls. *Geographical Analysis.* 25(2): 118–129.

OGC. 2014. Simple feature access, Part 1: common architecture, version 1.2.1 [online]. Open Geospatial Consortium. Available from: http://www.opengeospatial.org/standards/sfa [Accessed 2 Dec. 2014].

Openlayers. 2015. ol.geom.flat.simplify.douglasPeucker (http://bit.ly/1vLJvaq) [online]. Available from: http://openlayers.org/ [Accessed 14th May, 2015].

Opheim, H. 1982. Fast data reduction of a digitized curve. *Geo-Processing.* 2: 33–40.

Oracle Spatial. 2015. SDO_UTIL.SIMPLIFY. SDO_UTIL

Package (Utility) (http://bit.ly/1JTdbv3). [online] Available from: http://docs.oracle.com/cd/B28359_01/appdev.111/b28400/sdo_util.htm#SPATL1246/ [Accessed 14 May 2015]

Pallero, J.L.G. 2013. Robust line simplification on the plane. *Computers & Geosciences.* 61(0): 152159. doi: http://dx.doi.org/10.1016/j.cageo.2013.08.011

Poorten, P.M., S. Zhou, and C. Jones. 2002. Topologically-consistent map generalisation procedures and multi-scale spatial databases. In: M. Egenhofer and D. Mark, Eds. *Geographic Information Science.* Springer Berlin Heidelberg. 209–227.

PostGIS. 2015. ST_SimplifyPreserveTopology (http://bit.ly/1vhwBCg) [online]. Available from: http://postgis.net/docs/ [Accessed 14 May 2015].

Qingsheng, G., C. Brandenberger, and L. Hurni. 2002. A progressive line simplification algorithm. *Geo-spatial Information Science.* 5(3): 41–45. doi: 10.1007/BF02826387

Ramer, U. 1972. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing.* 1(3): 244–256.

Raposo, P. 2013. Scale-specific automated line simplification by vertex clustering on a hexagonal tessellation. *Cartography and Geographic Information Science.* 40(5): 427–443.

Reumann, K., and A.Witkam. 1974. Optimizing curve segmentation in computer graphics. Paper presented at the Proceedings of the International Computing Symposium.

Saalfeld, A. 1999. Topologically consistent line simplification with the Douglas-Peucker algorithm. *Cartography and Geographic Information Science.* 26(1): 718.

Saux, E. 2003. B-spline functions and wavelets for cartographic line generalization. *Cartography and Geographic Information Science.* 30(1): 33–50.

Shi, S., and M. Charlton. 2013. A new approach and procedure for generalising vector-based maps of real-world features. *GIScience & Remote Sensing.* 50(4): 473–482.

Stefanakis, E. 2012. Trajectory generalization under space constraints. ed. *The Proceedings of the 7th International Conference on Geographic Information Science.* 18–21.

Theodoridis, Y., D. Papadias, E. Stefanakis, and T. Sellis. 1998. Direction relations and two-dimensional range queries: optimisation techniques. *Data & Knowledge Engineering.* 27(3): 313–336.

Van Oosterom, P. 1991. The reactive-tree: a storage structure for a seamless, scaleless geographic database, Ed. AUTOCARTO-CONFERENCE. 393–393.

Visvalingam, M., and J.D. Whyatt. 1993. Line generalisation by repeated elimination of points. *Cartographic Journal.* 30(1): 46–51.

Vivid-Solutions. 2014. JTS topology suite. Available online at: http://www.vividsolutions.com/jts/JTSHome.htm

Warith, S. 2008. Contextually-based framework for improved data reduction in regional scale analytic element groundwater models. PhD Thesis. Available online at: http://gradworks.umi.com/33/20/3320437.html

Weibel, R. 1997. Generalization of spatial data: principles and selected algorithms. In: KREVELD, M., et al., Eds. *Algorithmic Foundations of Geographic Information Systems.* Springer Berlin Heidelberg. 99–152.

Wu, F., and H. Deng. 2003. Using genetic algorithms for solving problems in automated line simplification. *Acta Geodaetica Et Cartographic Sinica.* 4: 013.

Wu, S.-T., A.C.D. Silva, and M.R. Márquez. 2004. The Douglas-Peucker algorithm: sufficiency conditions for non-self-intersections. *Journal of the Brazilian Computer Society.* 9(3): 67–84.

Wu, S.T., and M.R.G. Márquez. 2003. A non-self-intersection Douglas-Peucker algorithm, Ed. Computer Graphics and Image Processing. SIBGRAPI 2003. XVI Brazilian Symposium on. 60–66.

Authors

Titus Tienaah, MScE, is a PhD candidate in the Department of Geodesy and Geomatics Engineering at the University of New Brunswick. His research focus includes spatial data structures, geographic information systems and high performance computing.

Emmanuel Stefanakis, PhD, PEng, is an Associate Professor in the Department of Geodesy and Geomatics Engineering at the University of New Brunswick. His teaching and research is focused on geospatial web and knowledge discovery from geographic data. He has over 60 articles published in international journals and conferences in Geomatics.

David Coleman, PhD, PEng, is a Professor of Geomatics Engineering at the University of New Brunswick and acted as Dean of UNB's Faculty of Engineering from 2003–2008 and again from 2009–2014. Dr. Coleman has authored over 150 publications and reports dealing with land information policy development, geomatics operations management, geographic information standards and spatial data infrastructure. ❑