Preserving Coincidence and Incidence Topologies in Saalfeld's Polyline Simplification Algorithm

Adler C. G. da Silva, Shin-Ting Wu

Departament of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (UNICAMP)
P.O. Box 6101, 13083-970 – Campinas, SP, Brazil

{acardoso,ting}@dca.fee.unicamp.br

Abstract. *In this paper, we firstly describe two topological configurations that are not considered by Saalfeld's polyline simplification algorithm: the coincidence topology, concerning the overlapping of two polylines or the overlapping of a feature point and a polyline, and the incidence topology, concerning the incidence of two polylines without having the incidence point represented as a common vertex. Afterwards, we suggest a simple modification in Saalfeld's algorithm for preserving these topologies. Finally, we give some results of our simplification procedure and compare them to the ones of Saalfeld's algorithm.*

1. Introduction

In Geographic Information Systems, polyline simplification is widely used to reduce digital map data information for the purpose of speeding up visualization. The simplification of a polyline is a technique which replaces the original linear feature by a less complex representation [Saalfeld 1999]. A variety of techniques has been presented by researchers in different contexts [Tobler 1964, Lang 1969, Reumann and Witkam 1974, Jenks 1981]. In cartography, the classic Ramer-Douglas-Peucker (RDP) algorithm [Ramer 1972, Douglas and Peucker 1973] is recognized as the one that delivers the best perceptual representations of the original line.

One important characteristic of a technique of simplification is to consider the features in the vicinity of the polyline. Since the simplified polyline may lie far apart from the original one, a city can appear inside a lake or changes its side with respect to a river or a political boundary. Such geometric peculiarities are related to the topology of a map. The RDP algorithm does not consider these properties and, hence, cannot preserve the original topology of most maps. Techniques like the RDP algorithm, which take the polyline *in vacuo* or in isolation [Saalfeld 1999], require some additional adjustments in a post-processing stage. Since these corrections do not make use of the original data, some conflicts cannot be removed.

Some algorithms modify a polyline *in suite* or in context, taking the relationships of the polyline with other nearby features into consideration during the adjustment process [de Berg and Kreveld 1995, Saalfeld 1999]. In these techniques, there is no need of post-processing stage, since the topology is preserved in the course of the simplification procedure. In [Saalfeld 1999], Saalfeld introduced a modification of the RDP algorithm, in order to ensure the topological equivalence between the original and simplified polylines. He assumes that the input maps have been partitioned into disjunct open subsets, which are not the case of most of digital maps in the known repository [DCW 1992]. Such situation appears frequently in maps in which rivers are part of the political boundaries, as illustrated in Figure 1. Notice that most part of the political borderlines of the State of Alagoas overlaps river lines and some river lines cross the state frontier, without the consistent insertion of a common vertex.

Figure 1. Map of the drainage network of the State of Alagoas.

It is common to encounter polylines that are described by non-disjunct sets:

- Two polylines, represented by two distinct sequences, with some coincident vertices, as depicted in Figure 2(a). We may re-partition this point set into five disjunct open polylines ($\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, and $\mathcal{P}_5$) and two points (p_1 and p_2), as shows Figure 2(b), to beside the redundancies.
- Two polylines, represented by two distinct sequences, with an incident point, which is not represented as a common vertex, as depicted in Figure 2(c). In this case, one alternative to remove the geometrical redundancies is to re-partition them into three disjunct open polylines ($\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$) and one point (p), as illustrates Figure 2(d).

For the purpose of distinguishing between these two types of geometrical redundancies, we introduce two concepts: (1) the coincidence topology, when either two polylines $\mathcal{P}_1$ and $\mathcal{P}_2$ coincides in one or more vertices, or a vertex of a polyline coincide with a feature point, and (2) the incidence topology, when an extreme vertex of a polyline $\mathcal{P}_2$ falls on a polyline $\mathcal{P}_1$, without the insertion of a common vertex in $\mathcal{P}_1$.

In this paper, we suggest a simple modification in Saalfeld's algorithm for preserving the coincidence and incidence topologies to handle the input data that contain geometrical redundancies. As our goal is for efficient visualization, we show that the identification of the so-called essential vertices suffices. We firstly present, in Section 2., in a very intuitive way, the concept of topological properties of a map represented by polylines and points. Then, in Section 3., we give a brief description of Saalfeld's polyline simplification algorithm. Next, in Section 4., we introduce our proposal for maintaining the original coincidence and incidence topologies. Afterwards, in Section 5., we compare

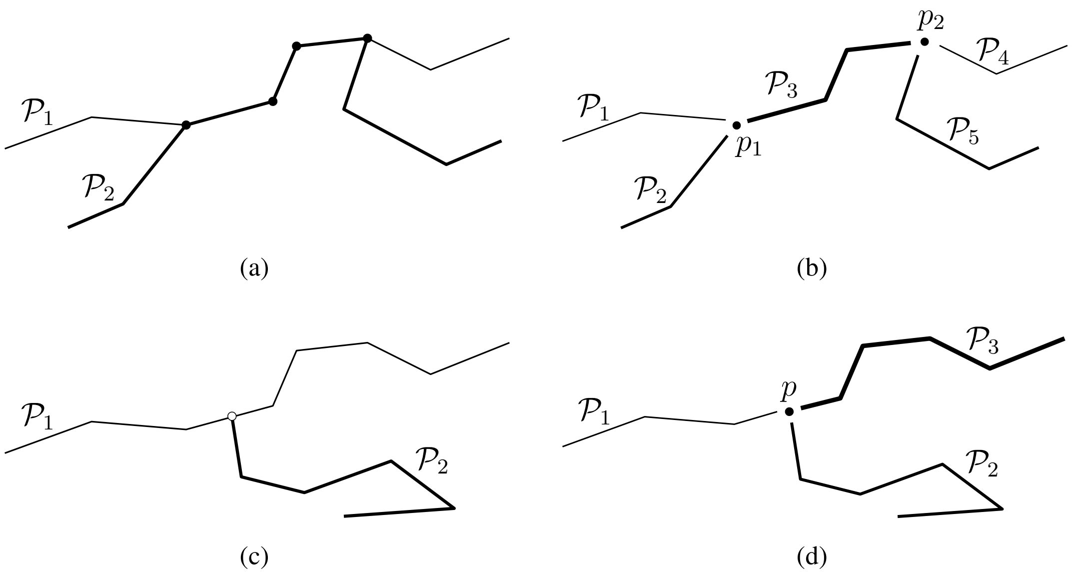


Figure 2. Geometrical redundancies and their respective correct topological representations: (a) the partially coincident polylines $\mathcal{P}_1$ and $\mathcal{P}_2$, and (b) their partitioning into the disjunct subsets $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, $\mathcal{P}_5$, p_1, and p_2; (c) the incident polylines $\mathcal{P}_1$ and$\mathcal{P}_2$, and (c) their partitioning into the disjunct subsets $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and p.

some simplified results of the modified version of Saalfeld's algorithm to the outcomes from the original Saalfeld's algorithm. Finally, in Section 6., we present some concluding remarks.

2. Topology in Maps

Topology deals with geometric properties where distance is not relevant. Among the topological characteristics we may mention the connectivity, the inclusion and the adjacency of points in a geometric figure. In the context of cartography, for instance, a map simplification which diminishes the sinuosity of a river or increases the size of a lake only alters the map geometry. On the other hand, a map simplification that produces self-intersections in a river or increases the size of a lake, such that it comprises a city that originally lies outside, does modify the map topology.

A topological property can be either intrinsic or relative. The intrinsic topology is related to the properties of the polyline itself, such as the self-intersection. Therefore, if a polyline has no self-intersection, it should keep its non-self-intersecting state after the simplifying process. Wu and Márques introduced a modified RDP algorithm, based on the concept of star-shaped polyline which generates non-self-intersecting simplified polylines [Wu and Márquez 2003]. This work was later improved to present some additional star-shaped polyline properties in [Wu et al. 2004].

The relative topology focuses on the topological relationship between two distinguishing map features. A map simplification alters the relative topology of features, when, for example, it originates intersections between two originally separating rivers or changes the side of a city with respect to a river. Saalfeld's algorithm only preserves the sidedness of a polyline with respect to its neighbouring features. There are, however, much more relative arrangements for a map consisting of polylines and points.

For two points, there are two possible arrangements: they either overlap or not. Between a point and a polyline, we may distinguish the following arrangements: the point lies either exactly over the polyline or in one of its sides. Among the variety of arrangements between two polylines $\mathcal{P}_1$ and $\mathcal{P}_2$, we may reduce them into four basic configurations: (1) $\mathcal{P}_1$ and $\mathcal{P}_2$ touch only on their extreme vertices (Figure 3(a)), as in a political borderline; (2) an extreme vertex of $\mathcal{P}_2$ touches an intermediate vertex $\mathcal{P}_1$ (Figure 3(b)), as in a river ramification; (3) $\mathcal{P}_1$ and $\mathcal{P}_2$ overlap on some intermediate vertices (Figure 3(c)), as in the superposition of a river and a political borderline; (4) $\mathcal{P}_1$ and $\mathcal{P}_2$ cross in some intermediate points that are not vertices (Figure 3(d)), as in a river road bridge; and (5) an extreme vertex of $\mathcal{P}_2$ falls on $\mathcal{P}_2$, without the insertion of a common vertex (Figure 3(e)).

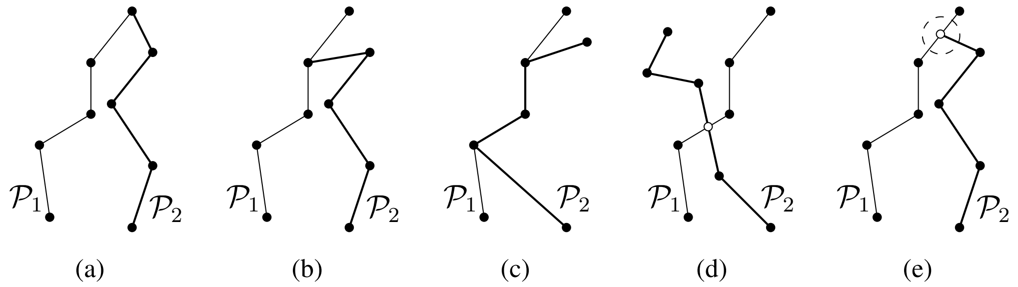


Figure 3. Basic arrangements between the polylines $\mathcal{P}_1$ and $\mathcal{P}_2$.

3. Saalfeld's Polyline Simplification Algorithm

Saalfeld's technique comprises two steps. The first step runs the classic RDP algorithm over the original polylines under the desired tolerance ϵ. The second step adjusts the topology of the simplified polyline, by adding more vertices until all sidedness incorrections are removed. Saalfeld proved the convergency of this method, by showing that, in the worst case, it will add all vertices and remount the original consistent polyline.

Let $\mathcal{P}$ be a polyline and $\mathcal{P}'$ the result of its simplification. Let $\mathcal{P}_{ij}$ be the subpolyline of $\mathcal{P}$ from the vertex v_i to the vertex v_j. Let e_{ij} be the result of the simplification of $\mathcal{P}_{ij}$ in $\mathcal{P}'$, as shown in Figure 4. Saalfeld proved that the features that lie between the line segment e_{ij} and the subpolyline $\mathcal{P}_{ij}$ (i.e., inside the complex polygon made by the edges of $\mathcal{P}_{ij}$ and e_{ij}) always lie in the "wrong" side with respect to $\mathcal{P}'$, as depicted in Figure 4.

To determine if a feature f lies inside the polygon bounded by the edges of $\mathcal{P}_{ij}$ and e_{ij}, Saalfeld made use of a simple algorithm based on the Jordan Curve Theorem. It calculates the number of crossings between a ray originating from f and the complex polygon made by $\mathcal{P}_{ij}$ and e_{ij}. The feature f is outside the polygon or in the "right" polyline side, when this number is even. Otherwise, it is inside the polygon or in the "wrong" polyline side. Saalfeld considered that a simplified polyline was topologically consistent to its original polyline, when every feature f was in the "right" polyline side.

The topology correction for the interval between vertices v_i and v_j is done recursively, by adding to the line segment e_{ij} the other vertices of subpolyline $\mathcal{P}_{ij}$, until all

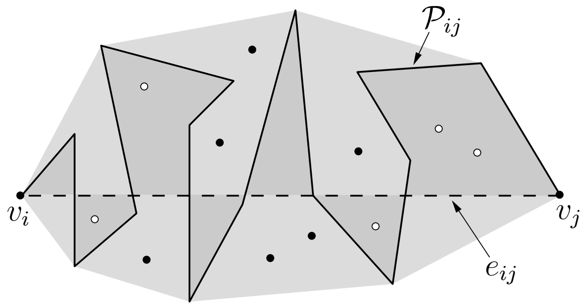


Figure 4. Analysis of the topological behavior of features that lies inside the polyline's convex hull.

features lie in the "right" poyline side. However, during this procedure, some features that are in the "right" polyline side in a step, can lose their correct sidedness in the next step. Saalfeld showed that only the features inside the convex hull of $\mathcal{P}_{ij}$ may change their sidedness, as depicted in Figure 4.

This procedure can be summarized, for each line segment e_{ij} of the simplified polyline $\mathcal{P}'$, in the following steps:

1. verify, among the features in the convex hull of the subpolyline $\mathcal{P}_{ij}$, if there is any between the line segment e_{ij} and $\mathcal{P}_{ij}$;
2. verify, among the vertices from v_{i+1} to v_{j-1} of the subpolyline $\mathcal{P}_{ij}$, if there is any farther from the line segment e_{ij} than the tolerance ϵ;
3. if step 1 or step 2 succeed, then add the farthest vertex v_k from the line segment e_{ij}, generating the line segments e_{ik} and e_{kj}; otherwise, stop;
4. calculate the convex hulls of the subpolylines $\mathcal{P}_{ik}$ and $\mathcal{P}_{kj}$;
5. go back to step 1 for the subpolylines $\mathcal{P}_{ik}$ and $\mathcal{P}_{kj}$.

For each subpolyline $\mathcal{P}_{ij}$, Saalfeld's algorithm maintains a list of all features inside its convex hull, as well as their sidedness state, i.e., "right" or "wrong" side. When the algorithm adds a vertex v_k, it verifies which features lie inside each one of the convex hulls of $\mathcal{P}_{ik}$ and $\mathcal{P}_{kj}$, and update their sidedness state. This update is accomplished in a very efficient way, as we can observe in Figure 5. Only the features (denoted by small squares) that lie inside the triangle $\triangle v_i v_j v_k$ change their sidedness state. Besides, some features (indicated by small triangles) that appear outside both convex hulls are discarded.

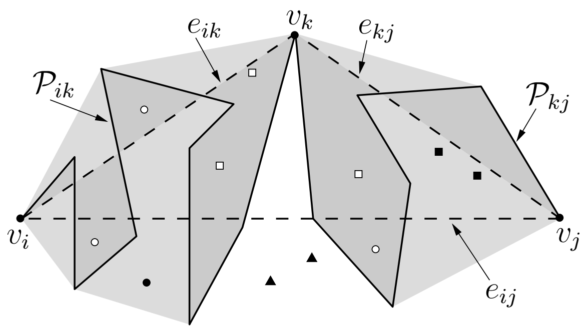


Figure 5. Update of the sidedness state of the features inside the convex hull of the subpolyline $\mathcal{P}_{ij}$ after adding the vertex v_k.

To compute the convex hull of the subpolylines, Saalfeld employed a enhanced version of Melkman's classic algorithm [Melkman 1987]. This enhanced version efficiently calculates the convex hulls of the two subsequent partitions $\mathcal{P}_{ik}$ and $\mathcal{P}_{kj}$ of a subpolyline $\mathcal{P}_{ij}$ based on its convex hull [Hershberger and Snoeyink 1992].

Once a vertex v_k is added to a simplifying line segment, it can interfere with other convex hulls and may lead to topological inconsistencies. Saalfeld's algorithm also takes into account the vertices added during the correction, by storing a list with the new added vertices and verifying if they lie inside of other convex hulls.

The polyline neighbouring features considered by Saalfeld's algorithm can be either external features, like cities and vertices of other polylines, or vertices of the polyline itself. Thus, Saalfeld unified the treatment for the intrinsic and the relative topologies. Nevertheless, since Saalfeld assumes that every feature is far from the polyline by a distance $\delta > 0$, his algorithm does not consider the coincidence and incidence topologies.

The map of Figure 6 is the outcome from Saalfeld's simplification of the map of Figure 1. Notice that many instances of coincidence topology inside the circumferences as well as of incidence topology inside rectangles were lost.

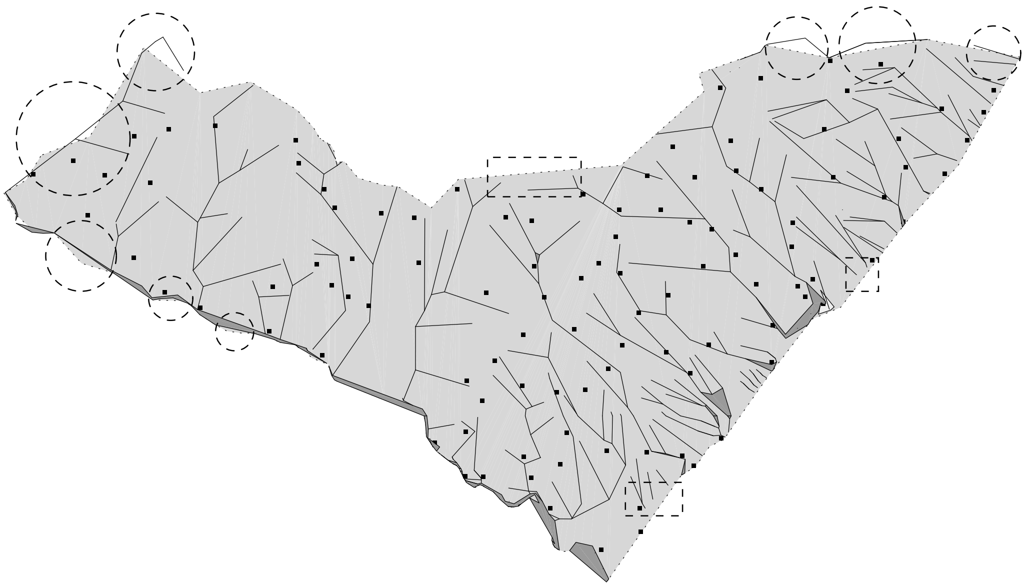


Figure 6. Outcome from Saalfeld's simplification of the map of Figure 1.

4. Preserving the Coincidence and Incidence Topologies

In order to preserve the coincidence topology, we propose to include a new criterion. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two polylines with a set of coincident vertices, whose simplified polylines are respectively $\mathcal{P}'_1$ and $\mathcal{P}'_2$. We ensure that if a coincident vertex appears in $\mathcal{P}_1$, then it will appear in $\mathcal{P}_2$, and vice-versa. We consider that two vertices of two distinguishing polylines are coincident if their computer discrete representations are exactly equal, so that a simple equality test suffices to perform their comparison. Note that we can also apply this criterion to a polyline and a isolated feature point, such as a city.

Figure 7(a) illustrates three distinct polylines $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, with some geometrically coincident vertices. Their simplification under a high tolerance ϵ results, respectively, in the polylines $\mathcal{P}'_1$, $\mathcal{P}'_2$, and $\mathcal{P}'_3$, as depicted in Figure 7(b). As we can see, the circled vertex was just added to respect the coincidence criterion, since it was already coincident in the original lines. Nevertheless, the coincidence topology was lost, because some important coincident vertices were not preserved in any of the simplified polylines. Thus, the coincidence criterion by itself does not suffice to maintain the coincidence topology.

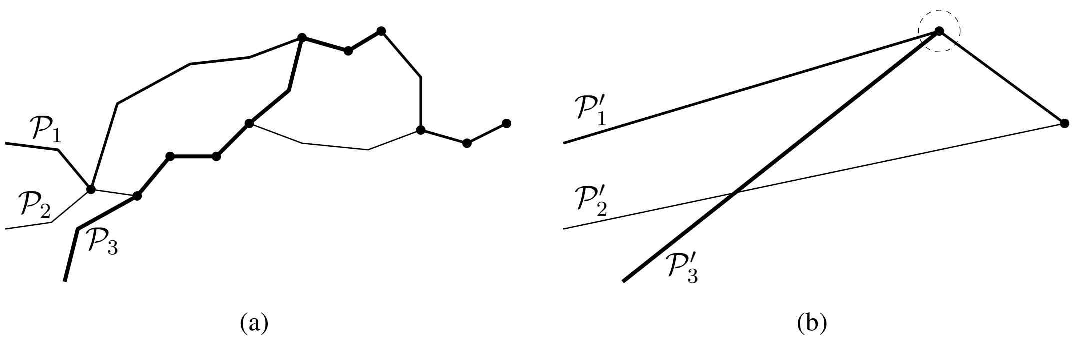


Figure 7. (a) Partially coincident polylines $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$; and (b) their topologically inconsistent outcomes from our simplification $\mathcal{P}'_1$, $\mathcal{P}'_2$, and $\mathcal{P}'_3$.

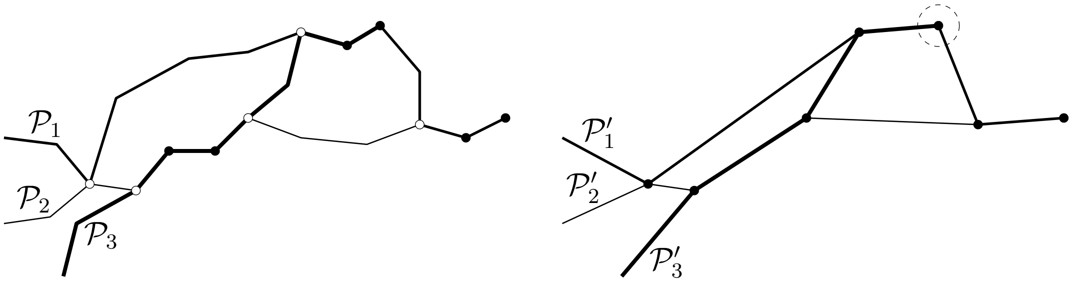
To remedy this problem, we introduce the concept of essential vertices. The essential vertices are the vertices which must appear in any simplified polyline, in order to ensure that the coincidence criterion suffices to preserve the coincidence topology. Essential vertices are coincident intermediate vertices, whose adjacent vertices are not both coincident, as illustrates Figure 8(a). Strictly speaking, let v_i and v_j be two coincident intermediate vertices of the polylines $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively. The vertices v_i and v_j are said to be essential if, and only if,

- $v_{i+1} \neq v_{j+1}$ or $v_{i-1} \neq v_{j-1}$; and
- $v_{i+1} \neq v_{j-1}$ or $v_{i-1} \neq v_{j+1}$.

Therefore, before starting the corrections, we must identify and insert all essential vertices in the simplified polylines. For identifying the essential vertices, we have to make a pairwise search among the subpolylines $\mathcal{P}_{ij}$ of the original polylines. Figure 8(a) illustrates the result of this search for the polylines $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$. Their simplification results are depicted in Figure 8(b). Once more, the circled vertex was just added to respect the coincidence criterion, since it was not classified as an essential vertex.

With regard to the conservation of incidence topology, we adopt a tolerance $\tau > 0$ to relax the incidence tests, named nearness tolerance. Although this approach does not succeed in all cases, a number of its subtle implications does not seem to visually affect the topological consistency of the map. We say that the polylines $\mathcal{P}_1$ and $\mathcal{P}_2$ are incident if an extreme vertex v of the polyline $\mathcal{P}_2$ lies at a distance $d < \tau$ from the polyline $\mathcal{P}_1$. The magnitude of τ must be related to the discrete approximation of the data. We say that the incidence topology is preserved if v and $\mathcal{P}_1$ still fulfill the nearness criterion after the procedure of simplification.

We devise a way to uniformly deal with the concepts of coincidence and incidence from the algorithmic point-of-view, by distinguishing three arrangements of a feature f with respect to a polyline $\mathcal{P}$:



(a) (b)

Figure 8. (a) Polylines $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$ with their essential vertices; and (b) their topologically consistent outcomes from the modified version of Saalfeld's simplification $\mathcal{P}'_1$, $\mathcal{P}'_2$, and $\mathcal{P}'_3$.

- "overlapping", when f is exactly equal to a vertex of $\mathcal{P}$;
- "near", when f lies at a distance $d < \tau$ from $\mathcal{P}$;
- "far", when f lies at a distance $d \geq \tau$ from $\mathcal{P}$.

Basically, our proposal is to ensure that, for each neighbouring feature f, its arrangement with respect to the simplified subpolyline $\mathcal{P}'_{ij}$ is the same as with respect to the original subpolyline $\mathcal{P}_{ij}$. Moreover, those features, which are "far" from the polyline, should lie in the "right" side of $\mathcal{P}'_{ij}$, as required by Saalfeld's criterion. Algorithm 1 gives an outline of our proposal.

```
Function checkConsistency(originalArrang, currentArrang: Arrangement;
                          side: Side): Boolean
Begin
   If originalArrang ≠ currentArrang then
      return FALSE;
   End if
   If currentArrang = FAR .and. side = WRONG then
      return FALSE;
   End if
   return TRUE;
End
```

Algorithm 1. Verification of topologically consistency of a neighbouring feature with respect to a polyline.

The original arrangement of a feature f with regard to the subpolyline $\mathcal{P}_{ij}$ can be computed with Algorithm 2. First of all, it verifies whether f "overlaps" any vertex of $\mathcal{P}_{ij}$ (line `04` to `08`). Next, it checks whether f lies "near" to any of the straight line segments of $\mathcal{P}_{ij}$ (line `09` to `13`), on the basis of the computation of the distance with the function distance (line `10`). Otherwise, the algorithm assumes that f lies "far" from $\mathcal{P}_{ij}$ (line `14`).

Our classification begins with respect to the roughest simplified polyline of $\mathcal{P}_{ij}$, which is the line segment e_{ij} (or $\overline{\mathcal{P}[i]\mathcal{P}[j]}$). The initial arrangement of a nearby feature f with respect to e_{ij} can be computed with Algorithm 3. This algorithm just verifies whether

```
Function originalArrangement(P: Point[]; i, j: Integer; f: Point; τ: Real): Arrangement
Var
   k: Integer;
Begin
   For k ← i to j do
      If f = P[k] then
         return OVERLAPPING;
      End if
   End for
   For k ← i to j − 1 do
      If distance(f, P[k]P[k + 1]) < τ then
         return NEAR;
      End if
   End for
   return FAR;
End
```

Algorithm 2. Computation of the original arrangement between the feature f and the subpolyline $\mathcal{P}_{ij}$ under the nearness tolerance τ.

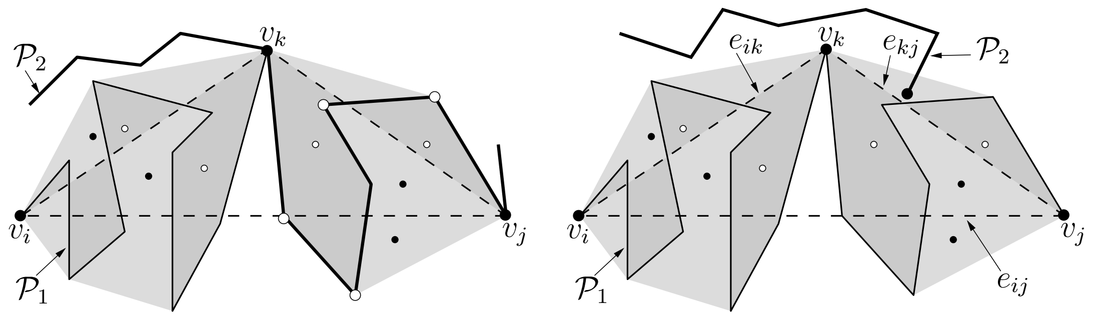
f "overlaps" v_i or v_j (line 02), or lies "near" to e_{ij} (line 05). Otherwise, it assumes that f lies "far" from e_{ij} (line 08).

```
Function initialArrangement(P: Point[]; i, j: Integer; f: Point; τ: Real): Arrangement
01 Begin
02    If f = P[i] .or. f = P[j] then
03       return OVERLAPPING;
04    End if
05    If distance(f, P[i]P[j]) < τ then
06       return NEAR;
07    End if
08    return FAR;
09 End
```

Algorithm 3. Computation of the initial arrangement between the feature f and the subpolyline $\mathcal{P}_{ij}$ under the nearness tolerance τ.

It remains to present a way to compute the arrangement of a feature f with respect to a simplified polyline $\mathcal{P}'_{ij}$ that is required by Algorithm 1. It is worth observing that this arrangement varies at each iteration, since new vertices are added to $\mathcal{P}'_{ij}$. When a vertex v_k is added to $\mathcal{P}'_{ij}$, the arrangement of f must be updated accordingly. Besides the case of Figure 5, two new cases may occur: (1) the feature may lie exactly over the vertex v_k (Figure 9(a)), and (2) the feature may lie near to the line segments e_{ik} and e_{kj} (Figure 9(b)).

If the feature f already "overlaps" v_i or v_j, this overlapping arrangement is preserved. Otherwise, after the insertion of the vertex v_k (or $\mathcal{P}[k]$), the current arrangement of f must be updated with respect to the simplified polyline $\mathcal{P}'_{ij}$, as illustrated in Algorithm 4. It firstly tests whether f "overlaps" v_k (line 02). If it does not, it tests whether f is "near" to $\overline{\mathcal{P}[i]\mathcal{P}[k]}$ or $\overline{\mathcal{P}[k]\mathcal{P}[j]}$ (line 05). Otherwise, it assumes that f is "far" from $\mathcal{P}'_{ij}$ (line 08).



(a) (b)

Figure 9. Update of the arrangement of the features after the insertion of a vertex v_k in (a) the coincidence case, and (b) the incidence case.

Function currentArragement($\mathcal{P}$: **Point**[]; i, j, k: **Integer**; f: **Point**; τ: **Real**): **Arrangement**

Begin
    If $f = \mathcal{P}[k]$ **then**
        return `OVERLAPPING`;
    End if
    If distance(f, $\overline{\mathcal{P}[i]\mathcal{P}[k]}$) < τ **.or.** distance(f, $\overline{\mathcal{P}[k]\mathcal{P}[j]}$) < τ **then**
        return `NEAR`;
    End if
    return `FAR`;
End

Algorithm 4. Update of the current arrangement between the feature f and the subpolyline $\mathcal{P}_{ij}$ under the nearness tolerance τ when the vertex $\mathcal{P}[k]$ is added.

Saalfeld's simplification algorithm may be further enhanced by integrating the Algorithms 1, 2, 3, and 4, according to the following scheme:

1. determine, for each feature lying inside and in the τ-vicinity of the convex hull of $\mathcal{P}_{ij}$, its original and initial arrangement with use of Algorithms 2 and 3, respectively;
2. update, at each iteration and for each feature lying inside and in the τ-vicinity of the convex hull of $\mathcal{P}_{ij}$, its current arrangement with respect to the current simplified subpolyline $\mathcal{P}'_{ij}$ with use of Algorithm 4;
3. check, at each iteration, whether the original topology of $\mathcal{P}_{ij}$ is consistent with the topology of its current simplified polyline $\mathcal{P}'_{ij}$ with help of Algorithm 1. If they are not consistent, go to step 2; otherwise, stop.

5. Results

For each map in this section, we compare the outcomes from the original and the modified versions of Saalfeld's simplification under a high tolerance ϵ and an appropriated nearness tolerance τ, in order to point out the enhancements achieved by our proposal.

The map of Figure 10 is the outcome from the modified version of Saalfeld's simplification of the map of Figure 1, which is result of overlapping of three layers: the ocean/political boundaries, the drainage network, and the populated places. Notice that the coincidence topology between the political borderlines and some river lines, as depicted in Figure 6, was correctly preserved, as well as the incidence topology between river lines extreme vertices and the political contour.

Figure 10. Result of the modified version of Saalfeld's simplification of the map of Figure 1.

Figure 11 illustrates the map consisting of three layers: the ocean/political boundaries, the road network, and the populated places of the State of Rio de Janeiro. The region in light gray represents the urban area of Rio de Janeiro City, part of whose boundary is the Bay of Guanabara's shorelines.

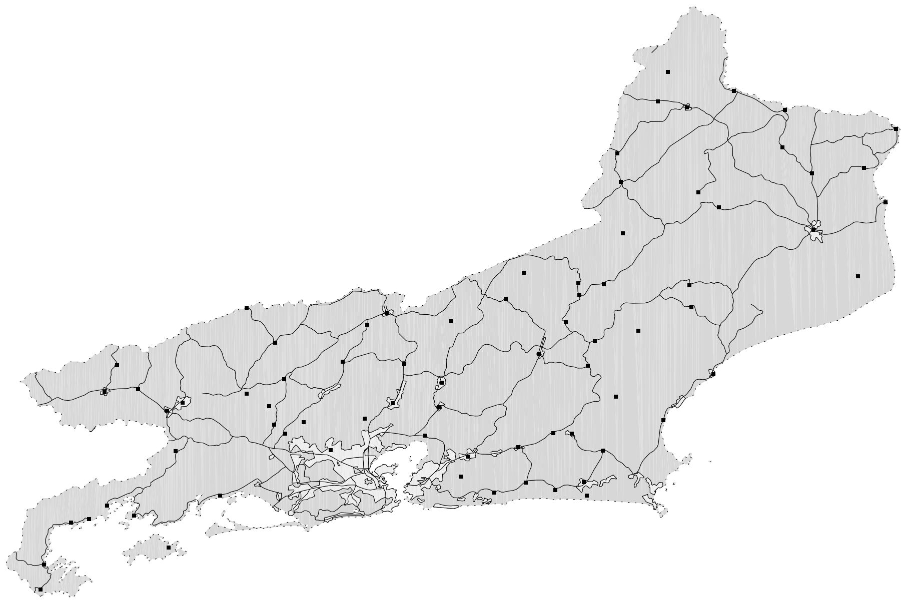


Figure 11. Map of the road network of the State of Rio de Janeiro.

The map of Figure 12 is the outcome from Saalfeld's simplification of the map of Figure 11. The coincidence topology was missed in the region pointed out by a circumference, which comprises the contour of Rio de Janeiro City and the Bay of Guanabara. Moreover, some road lines lost their incidence relationship with the political borderlines. They are pointed out by rectangles.

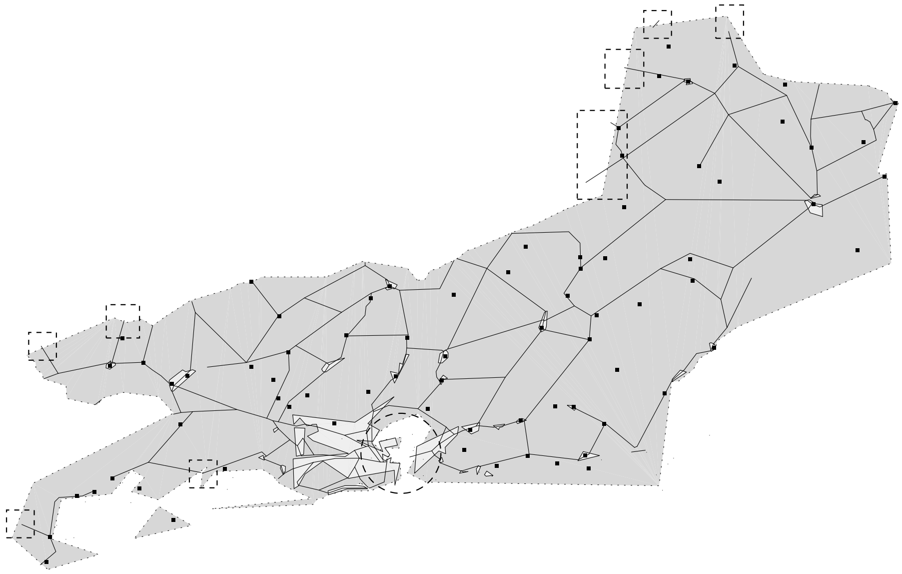


Figure 12. Result of Saalfeld's simplification of the map of Figure 11.

The map of Figure 13 is the outcome from the modified version of Saalfeld's simplification of the map of Figure 11. All the coincidence and incidence topology instances indicated in Figure 12 were preserved after the simplification.

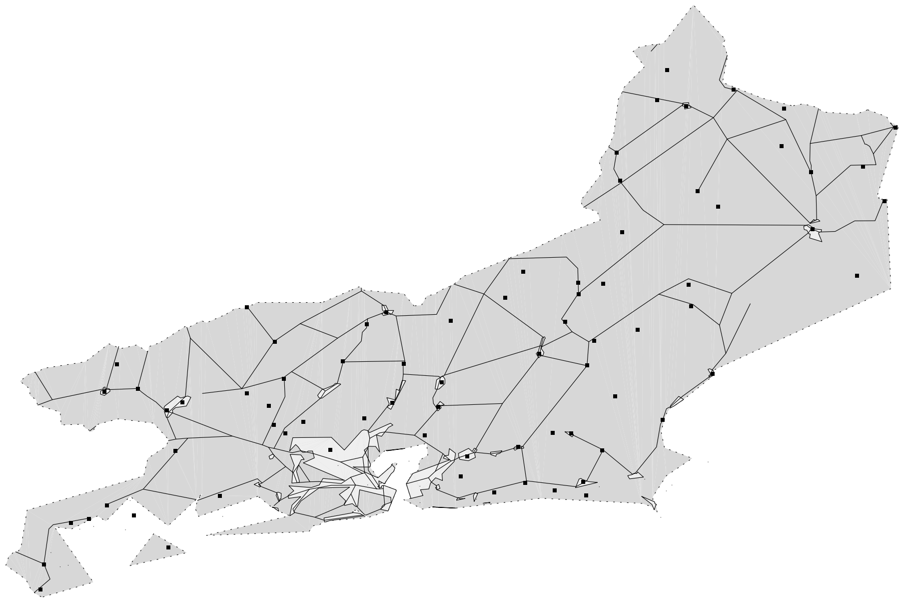


Figure 13. Result of the modified version of Saalfeld's simplification of the map of Figure 11.

Figure 14 presents a map of the State of Santa Catarina comprising three layers of geodata: the ocean/political boundaries, the drainage network, and the populated places.

As in the map of the State of Alagoas, it has lots of coincidence topology instances between river lines and the political borderlines.

Figure 14. Map of the drainage network of the State of Santa Catarina.

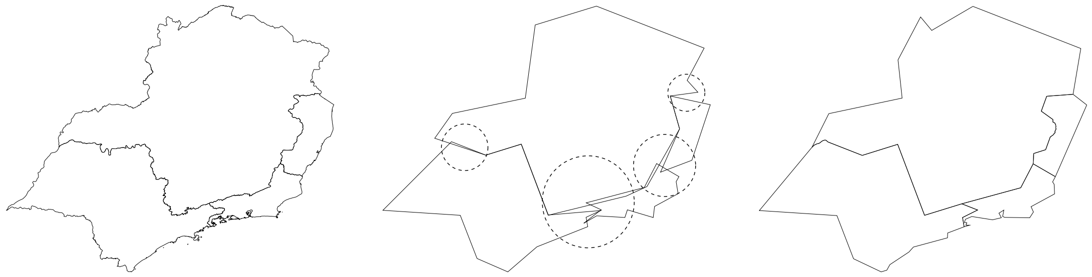
The map of Figure 15 is the outcome from Saalfeld's simplification of the map of Figure 14. Once more the original coincidence and incidence topologies were not preserved in regions pinpointed, respectively, by circumferences and rectangles.

Figure 15. Result of Saalfeld's simplification of the map of Figure 14.

The map of Figure 16 is the modified version of Saalfeld's simplification of the map of Figure 14. Our proposal again succeeds in preserving the coincidence and incidence topologies missed by Saalfeld's algorithm.

Figure 16. Result of the modified version of Saalfeld's simplification of the map of Figure 14.

Finally, we present outcomes from the simplification of the ocean/political boundaries of the States of the southeast region of Brazil, shown in Figure 17(a). In this case, we have redundancies as each state is represented by a polygon. Therefore, Saalfeld's simplification delivers several undesirable slivers and gaps, as shown in Figure 17(b), while its modified version can correctly handle the redundancies, as illustrated in Figure 17(c).



(a) (b) (c)

Figure 17. (a) The ocean/political boundaries of the southeast region of Brazil, and its representations resulting from (b) the original and (c) the modified versions of Saalfeld's simplification.

6. Concluding Remarks

In this paper, we showed that the map topology must be studied either in a intrinsic or relative context, and, in the latter, a diversity of topological arrangements of features must be considered. We also showed that Saalfeld's polyline simplification algorithm does not preserve the coincidence and incidence (without a common vertex) arrangements and introduced two concepts for distinguishing them: the coincidence and the incidence topologies.

In order to preserve the coincidence and incidence topologies in the data with geometrical redundancies, we proposed slight modifications of Saalfeld's algorithm: (1) we introduce a pre-processing phase for determining the essential vertices, and, (2) besides the sidedness criterion, we include the proximity criterion during the simplification procedure. The essential vertices fix the extreme vertices of the coincident subpolylines. This facilitates the handling of the coincidence topology. On the other hand, the proximity criterion, on its turn, allows us to deal with the incidence topology on the fly.

We applied our algorithm on maps of distinguishing data formats, all of them contain geometrical redundancies, and the visual quality of the results we obtained is satisfactory.

References

DCW (1992). Digital Chart of the World Server. Internet site `http://www.maproom.psu.edu/dcw/`.

de Berg and Kreveld (1995). A new approach to subdivision simplification. In *AutoCarto*, volume 12, pages 79–88.

Douglas, D. and Peucker, T. (1973). Algorithms for the reduction of the number of points required for represent a digitzed line or its caricature. *Canadian Cartographer*, 10(2):112–122.

Hershberger, J. and Snoeyink, J. (1992). Speeding up the Douglas-Peucker line simplification algorithm. In *The 5th International Symposium on Spatial Data Handling*, pages 134–143.

Jenks, G. F. (1981). Lines, computers and human frailties. In *Annals of the Association of American Geographers*, pages 1–10.

Lang, T. (1969). Rules for robot draughtsmen. *Geographical Magazine*, 22:50–51.

Melkman, A. A. (1987). On-line construction of the convex hull of a simple polyline. *Information Processing Letters*, 25:11–12.

Ramer, U. (1972). An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1:224–256.

Reumann, K. and Witkam, A. P. M. (1974). Optimizing curve segmentation in computer graphics. In *Proceedings of the International Computing Symposium*, pages 467–472.

Saalfeld, A. (1999). Topologically consistent line simplification with the Douglas-Peucker algorithm. *Cartography and Geographic Information Science*, 26(1):7–18.

Tobler, W. R. (1964). An experiment in the computer generalization of map. Technical report, Office of Naval Research, Geography Branch.

Wu, S.-T., da Silva, A. C. G., and Márquez, M. R. G. (2004). The Douglas-Peucker algorithm: sufficient conditions for non-self-intersecting. *Journal of the Brazilian Computer Society*, 1(1):1–1.

Wu, S.-T. and Márquez, M. R. G. (2003). A non-self-intersection Douglas-Peucker algorithm. In *The 16th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, pages 60–66. IEEE Press.