International Journal of Computational Geometry
& Applications
Vol. 24, No. 1 (2014) 61–86

DOI: 10.1142/S0218195914500034

TOPOLOGY-PRESERVING WATERMARKING OF VECTOR GRAPHICS

STEFAN HUBER
IST Austria
3400 Klosterneuburg, Austria
stefan.huber@ist.ac.at

MARTIN HELD*, PETER MEERWALD† and ROLAND KWITT‡
Universität Salzburg
FB Computerwissenschaften
5020 Salzburg, Austria
**held@cosy.sbg.ac.at*
†*pmeerw@cosy.sbg.ac.at*
‡*rkwitt@gmx.at*

Received 16 October 2013
Revised 29 April 2014
Communicated by J. S. B. Mitchell

Watermarking techniques for vector graphics dislocate vertices in order to embed imperceptible, yet detectable, statistical features into the input data. The embedding process may result in a change of the topology of the input data, e.g., by introducing self-intersections, which is undesirable or even disastrous for many applications. In this paper we present a watermarking framework for two-dimensional vector graphics that employs conventional watermarking techniques but still provides the guarantee that the topology of the input data is preserved. The geometric part of this framework computes so-called maximum perturbation regions (MPR) of vertices. We propose two efficient algorithms to compute MPRs based on Voronoi diagrams and constrained triangulations. Furthermore, we present two algorithms to conditionally correct the watermarked data in order to increase the watermark embedding capacity and still guarantee topological correctness. While we focus on the watermarking of input formed by straight-line segments, one of our approaches can also be extended to circular arcs. We conclude the paper by demonstrating and analyzing the applicability of our framework in conjunction with two well-known watermarking techniques.

Keywords: Watermarking; vector graphics; distortion constraints; topology.

*Work by Martin Held and Stefan Huber was supported by Austrian Science Fund (FWF): L367-N15 and P25816-N15.

1. Introduction

1.1. *Motivation*

Watermarking is a technology to enable copyright protection of digital assets. A standard approach to digital watermarking is to embed imperceptible, yet detectable, information into a digital signal that encodes the asset, see Cox *et al.*[9] As the watermark is only known to the embedder, copyright holders can use this technology to mark their content in order to prove ownership by being able to detect the embedded signal.

Most watermarking research has been directed towards techniques applicable to raster data (e.g., pictures and videos) and audio content. However, complex vector data in computer-aided design as well as maps and infrastructure data in geographic information systems, for instance, constitute equally valuable digital assets. We observe an increasing demand for suitable watermarking schemes, tailored to operate on these important assets.

When watermarking vector data, statistical features are imposed by dislocating vertices. Hence, one needs to introduce novel geometric requirements, while still demanding imperceptibility of the watermark and robustness against different kinds of attacks. For example, one needs to guarantee that watermarking does not introduce overlaps among rivers and streets in a map. Similarly, the pads of a printed circuit board should not overlap after the embedding. These geometric guarantees constitute an essential necessity when dealing with industrial datasets.

A separate strategy to deal with distortion of the original data due to watermarking is to resort to reversible watermarking techniques.[13,32] In addition to conveying information, a reversible watermark can be completely removed to recover the original data as long as no other modifications have been made. This is in contrast to the robustness requirement of watermarking in copyright protection applications on which we focus in this work.

1.2. *Prior work*

Surprisingly, copyright protection of vector data with distortion constraints has received little attention. Ohbuchi *et al.*[21] report, as a general rule and in lieu of a more refined algorithmic strategy, an acceptable error of $75cm$ in the real world on a 1:2 500-scale map. Obviously, this is a rather arbitrary choice which does not generalize well relative to diverse datasets. Doncel *et al.*[10] consider multiple polygonal chains sharing common vertices, such as the border of neighboring countries, and discuss how to preserve connectivity after watermarking. In Pu *et al.*[26] and Shao *et al.*,[29] methods are proposed that avoid perturbing certain vertices to preserve the visual appearance of the original data. In order to preserve the perceived shape of watermarked 2D vector data, Shao *et al.*[29] design an embedding method based on polygon simplification employing the Ramer-Douglas-Peucker algorithm.[11,27] Following a similar idea, Pu *et al.*[26] propose an alternative method based on normalized

meshes.[14] Both approaches explicitly avoid perturbing certain vertices to retain the visual appearance of the data set.

Pan *et al.*[23] present two algorithms for watermarking NURBS curves and surfaces. They claim that their experiments demonstrate that the embedding of their watermarks is shape-preserving, which should imply the preservation of the input topology. Similarly, Choi *et al.*[8] claim that the original polygonal mesh and the watermarked mesh are visually indistinguishable. However, both papers do not give formal guarantees that the input topology is indeed preserved. Recently, Bors and Luo[6] propose a 3D watermarking approach with minimal distortion while guaranteeing smoothness of the object surface. They point to related work aiming to preserve 3D model appearance after watermarking but do not consider topology changes.

Topology preservation has been studied in purely geometric settings, though. Estkowski and Mitchell[12] show that it is NP-hard to approximate an n-vertex polygonal subdivision without introducing Steiner points while preserving its topology such that its size is within a factor $n^{1/5-\delta}$ of optimal, for any $\delta > 0$. They also provide experimental evidence that their heuristic "Simple Detour" performs decently on real-world GIS data. Zhou and Bertolotto[5] employ an improved version of Saalfeld's modification[28] of the Ramer-Douglas-Peucker algorithm[11,27] to detect self-intersections of a simplified polygonal curve. In order to save time in real-time applications they suggest to pre-compute a sequence of approximations at different levels of detail during an offline phase. Khoshgozaran *et al.*[19] use a progressive transformation for transferring simplified polygonal data. Although not mentioned explicitly in their paper, topology preservation seems to come for free in their approach. However, it is unclear how to apply any of these line-simplification algorithms to the watermarking problem.

1.3. *Our contribution*

This work presents a watermarking framework with distortion constraints that guarantees to preserve the topology of the original input when a watermark is applied, which is in contrast to the best-effort approach of the aforementioned schemes. We focus on inputs formed by finite sets of straight-line segments that do not intersect, except at common endpoints. Such an input, which is commonly called a *planar straight-line graph* (PSLG), is a geometric graph obtained as the straight-line embedding of a planar graph comprising vertices and edges such that no vertex is isolated. Given a planar straight-line graph G as input, our watermarking framework preserves the topology of G by guaranteeing that

(T1) the numbers of vertices and edges,
(T2) all containment relations, and
(T3) all incidence orders at vertices

remain unchanged, and that

(T4) G remains planar.

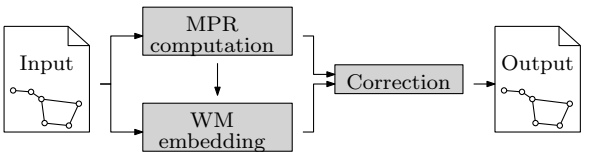


Fig. 1. Our watermarking framework.

The property (T2) guarantees that if an edge e of G is enclosed by a cycle of G then e remains enclosed by this cycle after the watermarked was applied.

Our framework consists of three steps, see Fig. 1. In the first step we compute a so-called *maximum perturbation region* (MPR) for every vertex such that T1–T4 can be guaranteed: If all vertices stay within their MPRs after watermarking then the input topology is guaranteed to be preserved. In the second step we apply a conventional watermarking scheme for vector data to G. Since this process need not respect T2–T4, in a final correction step we use the MPRs to correct the output of the watermarking step in order to guarantee T1–T4. However, some watermark techniques may already take the MPRs as additional information in order to produce output conforming with T1–T4. While the basic idea for preserving the input topology is simple, the computational challenge is to efficiently compute MPRs that are (i) large enough such that the correction step does not dampen the watermark embedding too severely and (ii) not too large such that T1–T4 would be violated. Our approach is general enough to work with any watermarking method that does not insert or remove vertices or edges of the input.

2. Maximum Perturbation Regions

We consider a planar straight-line graph $G = (V, E)$ with a vertex set $V = \{v_1, \ldots, v_n\}$, where $v_i \in \mathbb{R}^2$, and an edge set E. The process of embedding a watermark into G can be formulated as a mapping $\varphi\colon V \to \mathbb{R}^2$ that perturbs each vertex of G. We denote the perturbed graph by $G' = (V', E)$ with the perturbed vertex set $V' = \{\varphi(v_i) : v_i \in V\}$. The objective of the first step of our framework is to find *maximum perturbation regions* (MPRs) $R_1, \ldots, R_n$, with $v_i \in R_i \subset \mathbb{R}^2$, such that the following property holds: If $\varphi(v_i) \in R_i$ for all $1 \leq i \leq n$ then T1–T4 hold for G'. In other words, $R_1, \ldots, R_n$ are regions in which it is safe to perform perturbations without changing the input topology.

We will present two different techniques for the computation of MPRs. The first algorithm is based on (generalized) Voronoi diagrams, the other one is based on constrained triangulations. The advantage of the second technique is that computing constrained triangulations is simpler than computing Voronoi diagrams, and that it admits a generalization to $\mathbb{R}^3$ by employing a conforming tetrahedralization algorithm. On the other hand, as we discuss in the sequel, the Voronoi-based approach tends to yield bigger MPRs than the method based on triangulations and,

as a consequence, fewer and less severe corrections may be necessary after the watermark embedding. In addition, the Voronoi-based approach admits an extension to more general input primitives like circular arcs.

2.1. *MPRs based on Voronoi diagrams*

Voronoi diagrams of points and line segments are a fundamental concept in the field of computational geometry and have been intensively studied in the last decades. They lead to convenient solutions for a large number of geometric problems, including tool path planing, computing contour parallel offset curves or shape representation, to name only a few.

For the Voronoi-based approach to MPRs, we consider all vertices and straight-line edges of G as the set S of input sites. The Voronoi diagram $\mathcal{VD}(S)$ of S decomposes the plane into so-called Voronoi cells $\mathcal{VC}(s, S)$ around each input site $s \in S$. Roughly speaking, any point $p \in \mathcal{VC}(s, S)$ is closer to s than to any other site $s' \in S \setminus \{s\}$. (See the recent article by Held[16] for a precise definition and a survey of software for and applications of Voronoi diagrams of points, line segments and circular arcs.)

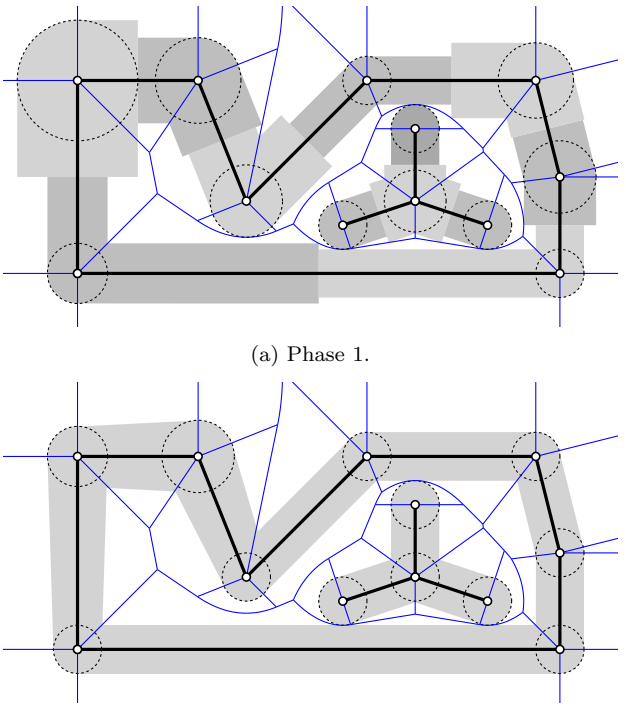
For an edge e of G we denote by $\varphi(e)$ the straight-line segment occupied by the perturbed counterpart in G'. Our MPR algorithm relies on the following observation: If $\varphi(e)$ does not intersect the Voronoi cells of edges non-adjacent to e, then G' remains planar as the Voronoi cells of different sites do not overlap. We denote by $|e|$ the length of the line segment e and define for any $r > 0$ and any line segment e the set $B(e, r) \subset \mathbb{R}^2$ as the rectangle with width $2r$ and length $|e|$ that has e as its center-line. Next, we denote by $D_v(r)$ the open disk with center v and radius r. Our MPR-algorithm consists of two phases:

Phase 1. For each vertex $v_i \in V$ of degree d_i, we consider the incident edges $e^i_1, \ldots, e^i_{d_i}$ and we denote by $\hat{e}^i_j$ the half of e^i_j that is incident to v_i. Then we compute for each vertex $v_i \in V$ the largest value $t_i \leq \min_{1 \leq j \leq d_i} |\hat{e}^i_j|$ such that

$$D_{v_i}(t_i) \cup \bigcup_{j=1}^{d_i} B(\hat{e}^i_j, t_i) \subseteq \mathcal{VC}(v_i, S) \cup \bigcup_{j=1}^{d_i} \mathcal{VC}(e^i_j, S). \tag{1}$$

Let $T(v_i) := D_{v_i}(t_i) \cup \bigcup_{j=1}^{d_i} B(\hat{e}^i_j, t_i)$, cf. Fig. 2(a). In order to compute $t_1, \ldots, t_n$ we first cut the Voronoi cell $\mathcal{VC}(e, S)$ of every edge $e \in E$, with $e = v_i v_j$, into two halves along the bisector of v_i and v_j, and insert the two points of intersection as Voronoi nodes. Every parabolic Voronoi edge is also split by a node at its apex. Then we traverse all halved Voronoi cells and compute the (non-zero) distances of the nodes of the halved Voronoi cells to their input sites.

Lemma 1. *The interiors of $T(v_i)$ and $T(v_j)$ do not overlap for different v_i and v_j.*



(a) Phase 1.

(b) Phase 2.

Fig. 2. The two phases of the Voronoi-based approach. The Voronoi diagram of the vertices and the edges is depicted in blue (color online). Phase 1: The regions $T(v_i)$ for vertices v_i are shaded in gray levels. The circles $D_{v_i}(t_i)$ are shown dashed. Phase 2: The MPRs are shown dashed. The union $\bigcup_{e \in E} H(e)$ of all hoses is shaded.

Proof. Assume to the contrary that there is a point p in the interior of $T(v_i) \cap T(v_j)$, for $v_i \neq v_j$. Since Voronoi cells do not overlap, Relation (1) implies that there is an edge $e = v_i v_j$ such that $p \in \mathcal{VC}(e, S)$. However, $T(v_i)$ and $T(v_j)$ cover different halves of $\mathcal{VC}(e, S)$ and p would lie in the interiors of both halves. □

Phase 2. For every vertex v_i we consider its adjacent vertices $v_1^i, \ldots, v_{d_i}^i$ and compute the value

$$r_i := \min\{t_{v_i}, t_{v_1^i}, \ldots, t_{v_{d_i}^i}\}. \tag{2}$$

Then we define the maximum perturbation region R_i of the vertex v_i as

$$R_i := D_{v_i}(r_i). \tag{3}$$

In Fig. 2(b), we illustrate all MPRs as dashed circles. For an edge $e \in E$ we call the area that is bounded by the MPRs of the incident vertices v_i, v_j of e and their (non-crossing) bi-tangents the *hose* $H(e)$ around e. In fact, $H(e)$ represents the valid area in which the perturbed edge $v_i' v_j'$ may lie. All hoses are shown as shaded

areas in Fig. 2(b). In order to avoid hoses that touch we can simply multiply all radii r_i by $1 - \epsilon$, for a small positive constant $\epsilon < 1$.

Lemma 2. *A hose $H(e)$ of an edge $e = v_i v_j$ is contained in $T(v_i) \cup T(v_j)$.*

Proof. Let $t := \min\{t_{v_i}, t_{v_j}\}$. Since $r_i \leq t$ and $r_j \leq t$, we get

$$\begin{aligned} H(e) &\subseteq D_{v_i}(t) \cup B(e,t) \cup D_{v_j}(t) \\ &\subseteq (D_{v_i}(t_i) \cup B(\hat{e}^i, t_i)) \cup (D_{v_j}(t_j) \cup B(\hat{e}^j, t_j)) \\ &\subseteq T(v_i) \cup T(v_j), \end{aligned}$$

which establishes the claim. □

Corollary 1. *$H(e_i)$ and $H(e_j)$ do not overlap if e_i and e_j have no common vertex.*

Theorem 1. *If the perturbation v_i' of the vertex $v_i \in V$ is constrained to R_i as defined in* (3)*, for $1 \leq i \leq n$, then T1–T4 are guaranteed for G'.*

Proof. Since no vertices or edges are added or removed, T1 is met. The hoses of one connected component of G are separated from all other hoses by the Voronoi diagram, thus enforcing T2. Similarly, due to the Voronoi diagram, the cyclic order of the hoses of edges incident upon a vertex is identical to the order of those edges and, therefore, T3 is guaranteed. Finally, Corollary 1 ensures T4. □

This algorithm assigns the MPRs in a *fair* manner in the following sense: If two hoses touch each other then they touch at the apex of a parabolic Voronoi edge. Hence, both hoses are equally wide as they are separated by the bisector of the input sites involved.

The multiple set inclusions in the proof of Lemma 2 suggest that the Voronoi-based approach does not necessarily determine the largest possible MPRs: Phase 2 is easy to implement but rather conservative. (For example, the MPR of the vertex in the upper left-hand corner of Fig. 2(b) could be chosen larger.) In theory, we could increase individual MPRs in an additional third phase as long as the perturbation of an edge $e = v_i v_j$ remains within $D_{v_i}(t) \cup B(e,t) \cup D_{v_j}(t)$, with $t := \min\{t_i, t_j\}$. Since the basic approach (as outlined above) yielded satisfying results in our experiments (Sec. 5), we did not study this extension in more detail.

Voronoi diagrams of n points, line segments and circular arcs can be computed in $O(n \log n)$ time in theory. In our implementation, we employ the software package VRONI[15] for computing Voronoi diagrams. The algorithm behind VRONI provides an expected runtime complexity of $O(n \log n)$ and has been proven to be fast and stable in practice. VRONI computes the Voronoi diagram of the *Carp* data set (24 134 vertices) depicted in Fig. 12 in about half a second on a standard PC. Once the Voronoi diagram is available, in Phase 1 we traverse the corresponding Voronoi cells and visit the neighbors of each vertex in Phase 2. Note that due to Euler's formula for planar graphs, the Voronoi diagram is of linear size (in n) and, hence,

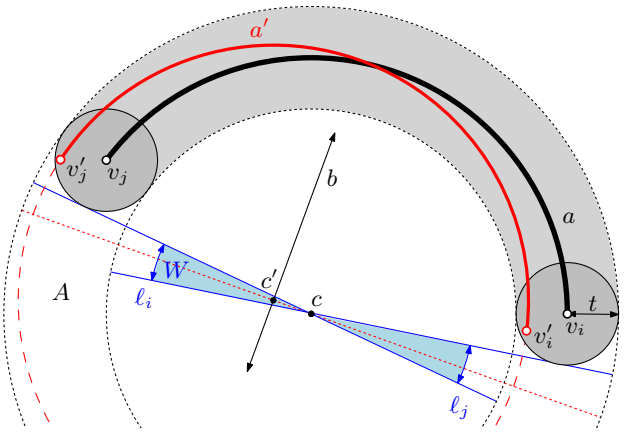


Fig. 3. Perturbing the endpoints of a circular arc.

both phases can be done in $O(n)$ time. To sum up, the time complexity of the entire MPR computation is in $O(n \log n)$. In practice, this allows to process large real-world data sets in a few seconds.

Since the ArcVroni[17] extension of Vroni supports the computation of Voronoi diagrams of points, line segments *and* circular arcs, our Voronoi-based approach to computing MPRs can be extended to input data that contains circular arcs. We only need to assume that every input arc is strictly less than a semi-circle.

In a nutshell, with such input, the endpoints of circular arcs are included in the signal into which the watermark is embedded. The arc's center is only adapted in order to match the perturbed endpoints. In the following we will define a hose $H(a)$ for a circular arc a with endpoints v_i and v_j and center c. Furthermore, we will explain how to obtain a new suitable center c' for the perturbed arc a'. The essential claim is that $a' \subset H(a)$, see Fig. 3.

As for line segments, we compute t_{v_i}, t_{v_j}, r_i and r_j by the procedure given above. In addition, we demand that t_{v_i} and t_{v_j} are small enough such that there exists a half-plane that contains the center c but neither $D_{v_i}(t)$ nor $D_{v_j}(t)$, where $t = \min\{t_{v_i}, t_{v_j}\}$.

We define the hose $H(a)$ of a as the Minkowski sum of a with a disk of radius t. (The Minkowski sum of two sets X and Y of position vectors is formed by adding each vector of A to each vector of B.) The claim is that we can always choose c' such that $a' \subset H(a)$ for any perturbation of v_i and v_j within their respective MPRs R_i and R_j. Let us consider the line l_i through c that is tangential to $D_{v_i}(t)$ and the line l_j through c that is tangential to $D_{v_j}(t)$. The lines l_i and l_j span a double-wedge W that avoids $H(a)$. The perturbed center c' must be equidistant to v_i' and v_j', i.e., it must lie on the bisector b between v_i' and v_j'.

Lemma 3. *For any $c' \in b \cap W$ it holds that $a' \subset H(a)$.*

Proof. First we note that $R_i, R_j \subset H(a)$ since $r_i, r_j \leq t$. Hence, $v_i' \in D_{v_i}(t)$ and $v_j' \in D_{v_j}(t)$. It suffices to show that a' is in the annulus A of thickness $2t$ that contains $H(a)$. We denote by C the supporting circle of a'. When we move a point p on C monotonically then the distance of p to c monotonically varies between a minimum and a maximum distance. Also note that minimum and maximum are attained where C is intersected by the supporting line of c and c'. By our choice of c' those intersection points are not contained in $H(a)$. Hence, as we move from v_i' to v_j' the distance to c changes monotonically. Therefore, as both v_i' and v_j' are contained in A, the entire arc a' is contained in A and consequently also in $H(a)$. □

In general, $b \cap W$ constitutes a line segment of positive length, from which we can choose c'. This flexibility allows us, for instance, to choose c' such that the curvature of a' is as close as possible to the radius of a.

Lemma 2 and Corollary 1 carry over literally for circular arcs. For Theorem 1 we need to guarantee that a circular arc and a line segment (or circular arc) that meet at a common endpoint v do not locally cross at v. To be precise, if we consider the tangential vectors of the arcs and segments meeting at v then the circular order must not change after perturbation. However, we can always shrink the MPRs of the endpoints of an arc a such that the tangential vector of a at v is bound by the tangential vectors of the two Voronoi edges between a and its neighboring line segment or arc at v. Hence, the Voronoi edges act as separators to avoid local intersections at v. For line segments this separation guarantee comes for free.

2.2. *MPRs based on triangulations*

Our second technique to compute MPRs starts with a constrained triangulation of G. A constrained triangulation T of G is a planar super graph of G that tessellates the convex hull of V into triangular faces, see Fig. 4. In other words, T is a triangulation of the vertices of G such that each edge of G is also an edge of T.

Let us denote by T' the graph that results from T after perturbing the vertices. The key observation in the triangulation-based approach is the following: If dislocating the vertices violates T2–T4 then at least one triangle in the triangulation has changed its orientation. That is, at least one triangle (v_i, v_j, v_k) switched from clockwise to counter-clockwise orientation, or vice versa. Hence, the objective is to restrict the perturbations of the vertices to MPRs such that no orientation of a triangle changes.

Consider a triangle Δ of T and denote by $I(\Delta)$ the radius of its incircle. If the vertices of Δ are dislocated by a distance of less than $I(\Delta)$, then the orientation of Δ remains the same. (See the proof of Thm. 2.) Hence, we compute the MPR R_i of the vertex $v_i \in V$ as

$$R_i := D_{v_i}(r_i), \tag{4}$$

where $r_i := \min_{1 \leq j \leq d_i} I(\Delta_j^i)$, with $\Delta_1^i, \ldots, \Delta_{d_i}^i$ denoting all triangles incident to v_i.

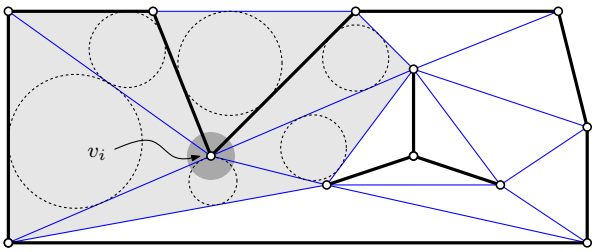


Fig. 4. The triangulation-based approach: R_i is defined by triangles incident to v_i.

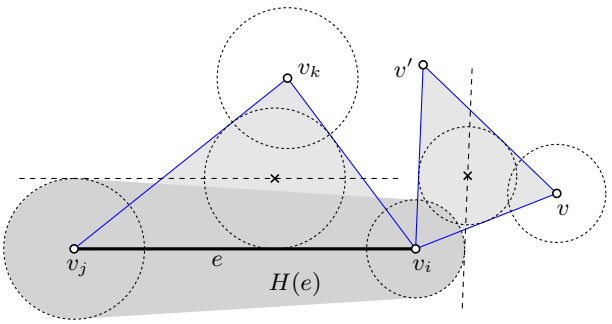


Fig. 5. The two cases of the proof of Theorem 2: The MPR of v_k and the MPR of v cannot intersect $H(e)$ since a separating line exists in both cases.

Since for each triangle Δ in T it holds that the MPRs of its vertices are disks with radii at most $I(\Delta)$, we get that all triangles in T preserve their orientations.

Theorem 2. *If the perturbation v_i' of the vertex $v_i \in V$ is constrained to R_i as defined in* (4)*, for $1 \leq i \leq n$, then T1–T4 are guaranteed for G'.*

Proof. As in the proof of Theorem 1, T1 is met since no vertices are added or removed. The cyclic order of hoses of edges incident upon a vertex v is given by the cyclic order of the triangulation edges incident upon v, thus ensuring T3. Since every hose includes its defining input edge in its interior, no containment relation can change and T2 holds unless two hoses would overlap (and T4 could be violated).

In order to establish T4, we start with proving that no MPR at a node v intersects a hose $H(e)$, with $e = v_iv_j$, unless $v = v_i$ or $v = v_j$. Let v_k be a vertex such that (v_i, v_j, v_k) forms a triangle Δ of T, see Fig. 5. By construction, the radii of the MPRs at v_i, v_j and v_k are less than or at most equal to the radius of $I(\Delta)$. Now consider a line ℓ through the center of the incircle of Δ that is parallel to e. We see that $H(e)$ lies completely within one closed half-plane induced by ℓ, while the MPR around v_k lies completely within the other open half-plane. We conclude that the MPR around v_k and the hose $H(e)$ do not overlap or touch if the vertices of e and

v_k form a triangle of T. In particular, the orientation of (v_i', v_j', v_k') is identical to the orientation of (v_i, v_j, v_k). By applying this argument repeatedly to all triangles incident upon v_k we learn that the MPR around v_k is completely contained in the interior of the union of all these triangles.

Hence, the only MPRs that could intersect $H(e)$ are MPRs around vertices of triangles incident upon v_i or v_j. So, consider a vertex v that is connected to v_i by an edge of T, but (v_i, v_j, v) do not form a triangle. Denote by v_k the vertex such that (v_i, v_j, v_k) forms a triangle and both v and v_k lie on the same side of the supporting line of e. There exists another vertex v' such that (v_i, v', v) forms a triangle Δ of T and at the same time the edge v_iv' lies in the angular sector spanned by v_iv_k and v_iv. Consider a line ℓ through the center of the incircle of Δ that is parallel to v_iv'. The MPR around v_i and the MPR around v_j lie in the same half-plane H bounded by ℓ. Since H is convex and $H(e)$ is the convex hull of the MPRs of v_i and v_j, $H(e)$ is contained in H too. On the other hand, the MPR of v lies on the other side of ℓ.

We conclude that there is no non-trivial intersection between an MPR and a hose. Can two hoses that do not share a common vertex intersect without involving an MPR in the intersection? Recall that the boundary of a hose consists of a circular arc of one of its MPRs, of a line segment that joins the two MPRs tangentially, of a circular arc of the second MPR, and of a second line segment. Hence, if two such hoses would intersect in an area that is disjoint from their four MPRs then the two edges defining the hoses would intersect, too. However, this is not possible since we build hoses only along edges of T, which is planar. Summarizing, we see that T4 is also guaranteed. □

We note that this algorithm works with any constrained triangulation T of G. However, in order to permit perturbations to robustly embed the watermark, we seek MPRs that are as large as possible. Hence, the question arises which triangulations of G have large incircles and subsequently large MPRs. Obviously, among all triangles with fixed circumcircle, the equilateral triangle has the largest incircle. This observation motivated us to employ the (constrained) Delaunay triangulation for our actual implementation of the above algorithm. However, maximizing the smallest incircle and maximizing the smallest inner angle of a triangle are not the same objectives. For example, a skinny triangle may be fine for us if it is just large enough and, in further consequence, contains still a large incircle. In Fig. 6 we show a simple example where the Delaunay triangulation does not yield the optimal solution. To the best of our knowledge, maximizing the smallest incircle has not attracted attention so far. However, Lambert[20] showed that the arithmetic mean of all incircle radii is indeed maximized by the Delaunay triangulation.

Guaranteed-quality triangulations that use Steiner vertices in order to guarantee a lower bound for the smallest angle in a triangulation are known, see for example Shewchuk.[30] Note that adding Steiner points to the triangulation does not hurt the correctness of our algorithm. However, successively adding Steiner vertices may increase the smallest angle, but, at some point, certainly does not enlarge the

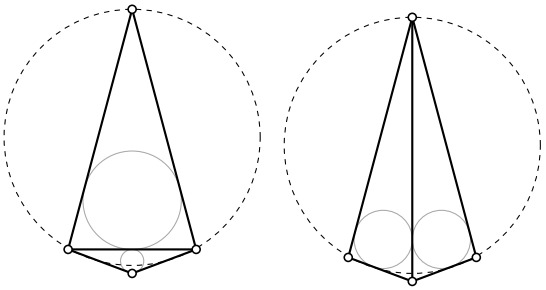


Fig. 6. Delaunay triangulations do not necessarily maximize the smallest incircle. Left: Delaunay triangulation. Right: Maximizing the smallest incircle radius.

smallest incircle any further. We implemented the following heuristic approach in order to enlarge the incircles: After computing the Delaunay triangulation of G, we determine a set of triangles Δ, where $I(\Delta)$ is by a factor of $f > 1$ larger when compared to its three neighboring triangles and insert Steiner points at the centers of their incircles. In practice, we observed that using $f = 1.5$ increases the average incircle radius by a few percent. However, our experiments showed that applying this refinement step multiple times does not lead to further improvement.

The constrained triangulation can be computed in $O(n \log n)$ time, in both theory and practice. Computing $r_1, \ldots, r_n$ can be done in $O(n)$ time as a triangulation contains $O(n)$ triangles. Hence, we end up with a total time complexity of $O(n \log n)$. For our C++ implementation of the triangulation-based algorithm we used the GNU Triangulated Surface (GTS) library,[25] which implements a semi-dynamic algorithm. Computing all MPRs on a dataset with 60 000 vertices and running one refinement step with $f = 1.5$ takes about a second on a mid-range computer.

The triangulation-based method can also be extended to $\mathbb{R}^3$, for instance if we want to embed a watermark on a polyhedron P with n vertices. The basic idea remains the same: We compute a tetrahedralization of P and determine for each vertex v the minimum r_v among all radii of the inscribed balls of incident tetrahedra. The MPR of v is then a ball with radius r_v. However, note that some polyhedra do not admit a tetrahedralization without employing Steiner points. Chazelle and Palios[7] showed that there exists a tetrahedralization of P that employs up to $O(n + r^2)$ Steiner vertices, where r denotes the number of reflex edges of P, and that it can be computed in $O(nr + r^2 \log r)$ time.

3. Correcting the Watermarked Data

In the final step of our framework we consider a watermarked graph G' and a list of MPRs $R_1, \ldots, R_n$ for the n vertices of G'. The graph G'' that is returned by the correction step must guarantee that the requirements T1–T4 are met. The brute-force strategy simply projects every vertex v_i' of G' that is not in R_i onto the boundary of R_i, see Fig. 7. This method takes only $O(n)$ time but may dislocate

more vertices than necessary. It is possible that G' already met the requirements T1–T4 despite some or even all vertices being outside their respective MPRs.

A more elaborate approach is to conditionally correct only those vertices that actually cause conflicts with T1–T4. The clear advantage of this approach is a better preservation of the watermark signal at a higher computational cost for the correction step. In the following we present two strategies that dislocate vertices of G' in order to meet the requirement T4. The requirement T1 is met by our choice of watermarking schemes. In order to guarantee T2, it suffices to enforce T4 and to require that at least one vertex of every connected component of G' is contained in its MPR. The requirement T3 can be met by checking the incidence orders at each vertex v and, if T3 is not met, correcting v and all its neighboring vertices. The above work can be easily done in $O(n)$ time. After this preprocessing we can concentrate on maintaining requirement T4.

3.1. *An $O(n^2)$ conditional correction strategy*

A naive brute-force algorithm by pairwise checks requires $O(n^3)$ time: When we find an intersection and correct the vertices of the edges involved, new intersections could be introduced, which requires a restart. We call an edge dirty if at least one of its vertices is not contained in its MPR. The following improved strategy keeps track of *dirty* edges.

The algorithm starts with building a queue Q that is initialized with all dirty edges of G'. As long as Q is not empty we dequeue an edge e. If e is still dirty, we check against all other edges of G' for an intersection. If an intersection with an edge e' is found we correct both e and e'. An edge is corrected by dislocating its vertices to the respective MPRs if necessary. For every dislocated vertex we need to re-insert the incident edges into Q. Note that each edge, if dirty, is inserted into Q at most three times: at initialization time and when each of its vertices is corrected. Hence, the total cost of the algorithm is in $O(kn)$ where $k \in O(n)$ denotes the number of dirty edges in G'.

For a practical implementation one would circumvent the intersection tests of e against all other edges by employing geometric hashing or more elaborate spatial decomposition data structures. In practical applications one expects that only a constant number of edges are in the neighborhood of an edge e that need to be checked for intersections. Hence, the algorithm is likely to run in linear time for real-world applications. In our experiments, we refer to this strategy as conditional MPR (cMPR).

3.2. *An $O(n \log n)$ average-case algorithm*

For an edge e in the watermarked graph G' we denote by $\psi(e)$ the line-segment that is occupied by e after correcting its vertices. (Hence, $\psi(e) = e$ for non-dirty edges e.) Our second conditional correction algorithm consists of two phases. In the first phase we build a binary relation D on the edges of G' that conveys the following

information: If we correct edge e_1 then we also need to correct e_2 as $\psi(e_1)$ and e_2 intersect, i.e., $(e_1, e_2) \in D \ :\Leftrightarrow \psi(e_1) \cap e_2 \neq \emptyset$. We may interpret D as a directed graph with the edges of G' as vertex set. (Note that D may contain cycles.) In the second phase, we find intersections of edges e_1 and e_2 of G', correct e_1 and e_2 and each edge that can be reached within D from e_1 and e_2.

The algorithm starts with obtaining the graph G'' from G' by correcting all vertices of G', which takes $O(n)$ time. We denote by E' the set of edges of G' and by E'' the edges of G''. Then we determine the set I of intersecting pairs of edges in $E' \cup E''$. (We ignore trivial intersections among adjacent edges, though.) This can be one in $O(n \log n + m)$ time and $O(n)$ space due to Balaban,[1] with m denoting the size of I. Next we iterate over I and build the dependency graph D (as an adjacency list) and a Boolean array B ('bad edges') that indicates for an edge of G' whether it needs to be corrected. This concludes the first phase.

For the second phase we create a Boolean array C ('clean edges') that saves for each edge in G' whether it is non-dirty; this takes $O(n)$ time. Then we loop over all edges e of G' and check whether $B[e] \wedge \neg C[e]$ is true. If the condition holds we start a depth-first search from e and correct every edge e_2 of G' that can be reached within D and for which $C[e_2]$ is not set. Note that the total time cost of this step is in $O(n + m)$, as D is given as an adjacency list. Hence, the overall complexity is in $O(n \log n + m)$.

For an actual implementation we would use the Bentley-Ottmann algorithm[4] in order to compute I, which is simple to implement but has a slightly worse time complexity of $O((n + m) \log n)$. Also note that in practice one expects that m is close to linear. Under this assumption the algorithm runs in $O(n \log n)$ time even if we have to correct $O(n)$ edges, which is in contrast to the previous algorithm.

4. Watermarking

For image raster data, Podilchuk and Zeng[24] pioneered the concept of perceptual shaping of the watermark according to the just-noticeable-difference (JND) value for each coefficient. The JND gives the approximate amount of modification a coefficient can tolerate before the change becomes visible according to a perceptual model. Perceptual models are often formulated in a transform domain such as the discrete cosine transformation[33] (DCT) or discrete wavelet transform[3] (DWT) domain.

Watermark embedding is either performed directly in the coordinate domain,[26,29] or in some transform domain: Ohbuchi *et al.*[21] utilize the mesh-spectral domain, Solachidis and Pitas[31] propose the complex discrete Fourier transform (DFT) domain due to its invariance against a number of attacks (translation, rotation, scaling). An advantage of transform-domain approaches is that selection of mid-frequency coefficients for watermarking, guided by a perceptual model, provides a convenient way to embed information in a significant portion of the host data without causing severe, noticeable perturbation. For transform-domain watermarking methods, the MPR distortion constraint has to be imposed *after* embedding, see

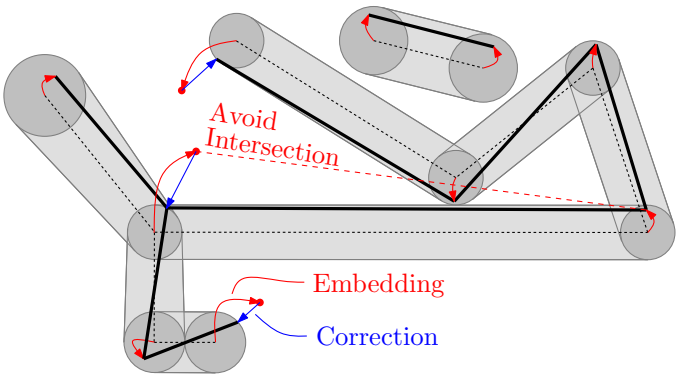


Fig. 7. Correcting the watermarked vertex data subject to the MPR distortion constraint.

Fig. 1. Vector data watermarking techniques operating in the coordinate domain can incorporate the MPR constraint directly in the ways indicated by Podilchuk and Zeng,[24] as we will see in Sec. 4.2.

In the following we discuss two exemplary watermarking schemes for vector data. The first algorithm operates in the frequency domain, the second in the coordinate domain. Both watermarking schemes have in common that they operate on polygonal chains, i.e., a sequence of vertices connected by edges. Hence, as a first step we extract independent paths from our input graph G, which are passed to the watermarking algorithm as a sequence of vertices. For this reason we consider all the paths $(v_{i_1}, \ldots, v_{i_k})$ in G with v_{i_1} and v_{i_k} being vertices of degree one, three or higher and v_{i_j}, with $1 < j < k$, being degree-two vertices. All paths together cover the entire edge set of G and no two paths share a common edge. From now on we consider a sequence of points in $\mathbb{R}^2$ as the input of a watermarking algorithm.

4.1. *Frequency-domain watermarking*

Let $\mathbf{v} = (v_1, \ldots, v_k)$ denote a sequence of k points, a polygonal chain. Each point $v_j \in \mathbb{R}^2$, with $1 \leq j \leq k$, is interpreted as a point x_j in the Gaussian plane. Hence, we obtain from $\mathbf{v}$ a complex signal $\mathbf{x} = (x_1, \ldots, x_k)$; we adhere to the notation introduced by Solachidis and Pitas.[31]

Using the well-known approach to watermarking vector graphics based on Fourier descriptors[10,31] as an example, a multiplicative spread-spectrum watermark $\mathbf{w}$ can be applied on selected (indicated by an overhead $\sim$) complex DFT coefficient magnitudes $|\tilde{\mathbf{x}}|$ of signal $\mathbf{x}$,

$$|\tilde{x}_j|' := |\tilde{x}_j| \cdot (1 + \alpha w_j), \tag{5}$$

where $\alpha > 0$ is the embedding strength and $w_j \in \{-1, 1\}$ with equal probability generated by a pseudo-random number generator seeded with a secret key K. Knowledge of the secret K enables the legitimate owner of the copyrighted work

to detect the watermark and thus prove ownership. Watermarking the DFT magnitudes provides invariance to geometric operations such as scaling, translation, rotation, change in traversal starting index, and mirroring.

Given a received polygonal chain $\mathbf{z}$, a blind watermark detector has to decide without reference to the original data between the two hypothesis

$$\begin{aligned} \mathcal{H}_0: &\quad \text{no/other watermark present in } \mathbf{z} \\ \mathcal{H}_1: &\quad \mathbf{z} \text{ is watermarked with } \mathbf{w}. \end{aligned} \tag{6}$$

In Solachidis and Pitas,[31] detection using linear correlation (LC) with test statistic $\rho_{LC} = \frac{1}{k}\sum_{j=1}^{k} |\tilde{z}_j| w_j$ was proposed. Doncel *et al.*[10] improved the detector by noting that the DFT coefficient magnitudes can be accurately modeled by a Rayleigh distribution with parameter $\lambda > 0$. The derived (estimate-and-plug[18]) likelihood-ratio test (LRT) conditioned on the host signal model decides $\mathcal{H}_1$ in case

$$\rho_{LRT} = \sum_{j=1}^{k} |\tilde{z}_j|^2 \frac{(1+\alpha w_j)^2 - 1}{2\hat{\lambda}_k^2 (1+\alpha w_j)^2} > T_\rho \tag{7}$$

where $\hat{\lambda}_j = \sqrt{\frac{1}{2(2p+1)}\sum_{l=j-p}^{j+p} |\tilde{z}_l|^2}$ is the maximum likelihood estimate of the Rayleigh distribution parameter within a sliding window of size $2p+1$ and T_ρ denotes the detection threshold (see Sec. 5). It can be argued that according to the Central-Limit theorem both detection statistics follow a Normal distribution for reasonably large data size, say $k > 1000$.

Due to MPR correction after watermark embedding, we expect that the watermark power is partially damped. MPR correction can be interpreted as a noise source on the watermark signal. Table 1 lists the number of watermarked vertices and number of corrections applied using the Voronoi-based MPRs for several data sets. In Sec. 5, we assess this degradation experimentally using the detectors presented above.

4.2. *Coordinate-domain watermarking*

In this section, we examine a coordinate domain (CD) watermarking approach put forward by Shao *et al.*[29] which is designed to preserve the 2D shapes of the vector data.

Given a polygonal chain $\mathbf{v} = (v_1, \ldots, v_k)$ composed of vertices $v_j \in \mathbb{R}^2$, the method first determines a sequence of m *feature* vertices $v_{\star_1}, \ldots, v_{\star_m}$, where $1 = \star_1 < \ldots < \star_m = k$, by means of the Douglas-Peucker algorithm.[11] (This algorithm is widely used for curve simplification.) These vertices $v_{\star_i}$ can be connected by a sequence of $m-1$ line segments $l_1, \ldots, l_{m-1}$, thereby representing a simplified version of the original polygonal data. Hence, line segment l_i connects feature vertex $v_{\star_i}$ with $v_{\star_{i+1}}$. The orthogonal distance from each *non-feature* vertex v_j, located between two feature vertices $v_{\star_i}$ and $v_{\star_{i+1}}$, to its corresponding line segment l_i is then computed, see Fig. 8. There are $k-m$ distances d_j, with $j \in \{1, \ldots, k\} \backslash \{\star_1, \ldots, \star_m\}$,

Table 1. Number of watermarked and corrected vertices after frequency domain embedding (Voronoi MPR).

Data set	Vertices	Water-marked Vertices	Corrected Vertices			
			$\alpha = 0.5$		$\alpha = 0.25$	
			MPR	cMPR	MPR	cMPR
Carp	24134	4890	801	13	1589	79
Chaffinch	23203	634	79	4	171	24
Hippopotamus	47378	12116	1265	48	2802	137
Pigeon	10523	2452	153	4	330	23
Chimpanzee	17296	779	80	0	186	0
Owl	29457	3391	470	13	987	41
Catbird	16353	1908	97	0	228	3

which constitute the host signal vector $\mathbf{d}$ to be modified by the watermark embedder. The distance vector $\mathbf{d}$ is pseudo-randomly partitioned into two vectors $\mathbf{d}^A$, $\mathbf{d}^B$ with roughly the same number of elements. This pseudo-random assignment of each distance d_j to either $\mathbf{d}^A$ or $\mathbf{d}^B$ is the secret key K, which enables copyright claims. The two sets share the same statistical properties, i.e. their sample mean and variance is approximately the same. The watermark embedding procedure now changes the variance of one set by multiplying each distance value with a factor $T_{DP} < \beta < 1$ while leaving the other set unchanged:

$$d'_j = \beta \cdot d_j \qquad \forall d_j \in \mathbf{d}^B. \tag{8}$$

The watermarked distance vector $\mathbf{d}'$ is finally applied to the non-feature vertices, i.e. approximately half of them are slightly moved towards their corresponding line segments defined by the circumjacent feature vectors. The embedding factor β is subject to a constraint depending on the threshold T_{DP} used in the Douglas-Peucker algorithm (see Shao *et al.*[29] for details); lower values of β relate to stronger marking. Only polygonal chains with a certain number of vertices ($k \geq 600$) are selected for watermarking.

Note that feature vertices determined by the algorithm of Douglas-Peucker constitute a low-resolution representation of a polygonal chain. By only watermarking the detail information, i.e. the non-feature vertices, the overall shape of the 2D data is preserved. A similar multi-resolution representation is employed by Pu *et al.*[26] with the help of normalized meshes[14] for watermarking the perceptually insignificant signal components.

The MPR constraint can be directly incorporated in the embedding algorithm illustrated in Fig. 8. In case the distance $d'_j = \beta \cdot d_j$ would displace the vertex v'_j outside its MPR (not shown in the figure), a corrected distance d''_j is computed

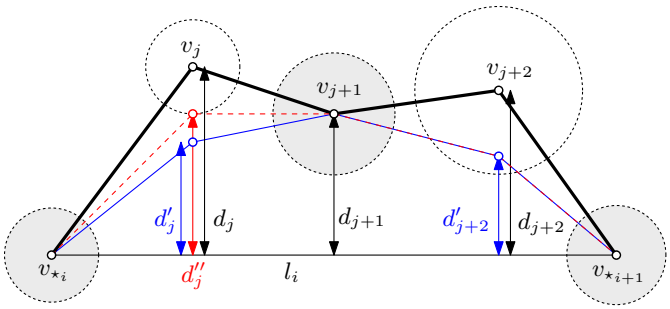


Fig. 8. Watermark embedding with MPR constraint in the coordinate domain. Vertices with white MPRs are dislocated by the watermark embedding. The blue line represents the watermarked vertex sequence without correction. The red dashed line represents the corrected vertex sequence (color online).

which places the watermarked vertex on the MPR boundary (as shown). Note that no correction is necessary if the MPR of v_j intersects l_i.

On the detection side, the possibly altered distance sets $\mathbf{e}^A$ and $\mathbf{e}^B$ are derived from the received vertices in the way described above (using the same pseudo-random assignment) and the variance within each set is computed. The watermark detection statistic is given by

$$\rho_{CD} = \frac{\operatorname{var} \mathbf{e}^B}{\operatorname{var} \mathbf{e}^A}. \tag{9}$$

In case a watermark is embedded, the ratio between the variances is close to β^2, otherwise close to 1 following the detection approach of Shao *et al.*[29] Again, the detection statistic adheres to a Gaussian law for sufficiently large sets $\mathbf{e}^A$, $\mathbf{e}^B$.

The number of watermarked and corrected vertices for the coordinate domain embedding approach is shown in Table 2 using the MPRs derived with the Voronoi method.

5. Experimental Results

Vector images used for this work are freely available and can be downloaded from the Internet[a] in SVG format. Python source code for the watermarking schemes is also available online.[b] Experiments were performed on a number of data sets of different size and type. Due to limited space, we can present only two representative examples here, see Figs. 10 and 12.

Figure 12(a) shows the watermarked *Carp* vector graphics consisting of 24 134 vertices in 836 polygonal chains. The watermarking algorithm based on Solachidis and Pitas[31] and Doncel *et al.*[10] selects polygonal chains with 400 or more vertices

[a]http://openclipart.org, http://openstreetmap.org.
[b]http://www.wavelab.at/sources.

Table 2. Number of watermarked and corrected vertices after coordinate domain embedding (Voronoi MPR).

Data set	Vertices	Water-marked Vertices	Corrected Vertices			
			$\beta = 0.8$		$\beta = 0.6$	
			MPR	cMPR	MPR	cMPR
Carp	24134	1965	147	0	320	0
Chaffinch	23203	3085	154	5	380	2
Hippopotamus	47378	8020	635	34	1401	25
Pigeon	10523	2479	84	0	216	0
Chimpanzee	17296	3706	187	0	491	0
Owl	29457	5744	537	4	1235	3
Catbird	16353	2755	79	1	234	0

and hence modifies 4 890 vertices by embedding with strength $\alpha = 0.5$. Eventually, 1 589 vertices were subjected to MPR correction. Table 1 provides embedding results on six additional data sets. For each data set, we list the number of watermarked vertices and the number of corrected vertices for the MPR and cMPR geometrical distortion constraint when embedding with strength $\alpha = 0.25$ and $\alpha = 0.5$.

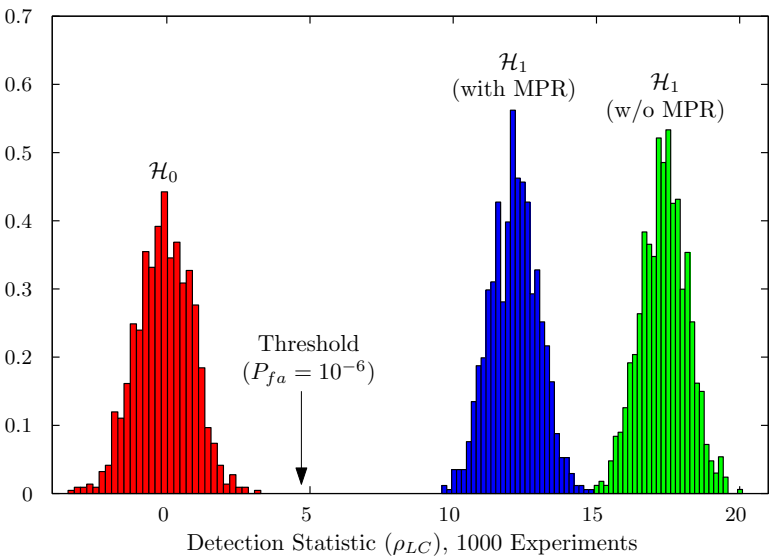


Fig. 9. Detection statistic of ρ_{LC} under $\mathcal{H}_0$ and $\mathcal{H}_1$ (with and without MPR correction) on *Carp* with $\alpha = 0.5$.

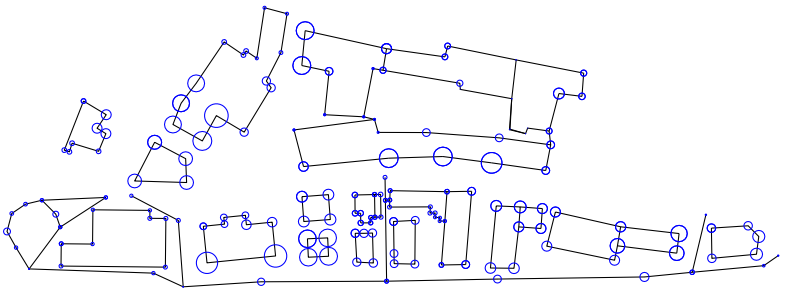


Fig. 10. MPRs on a GIS data set of a portion of the city of Salzburg.

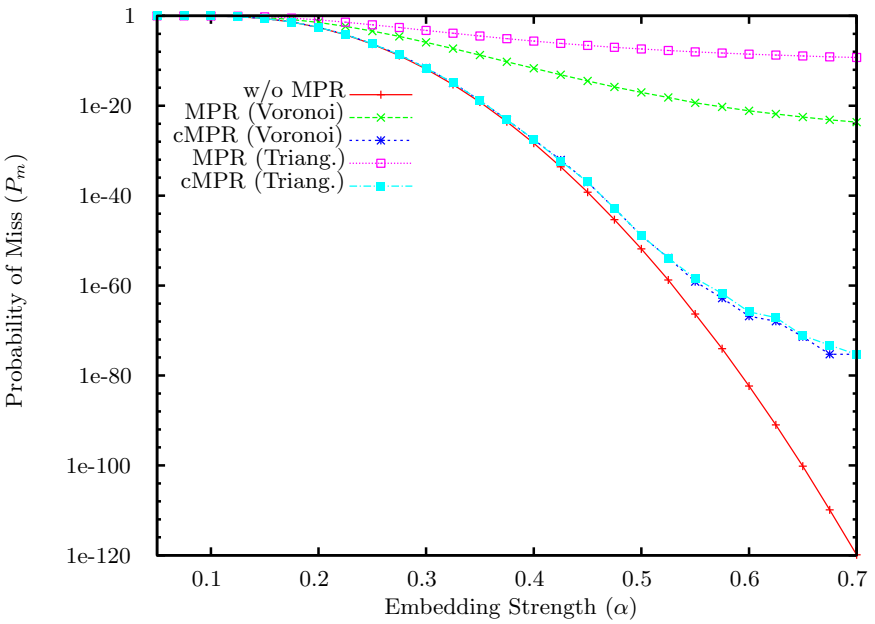
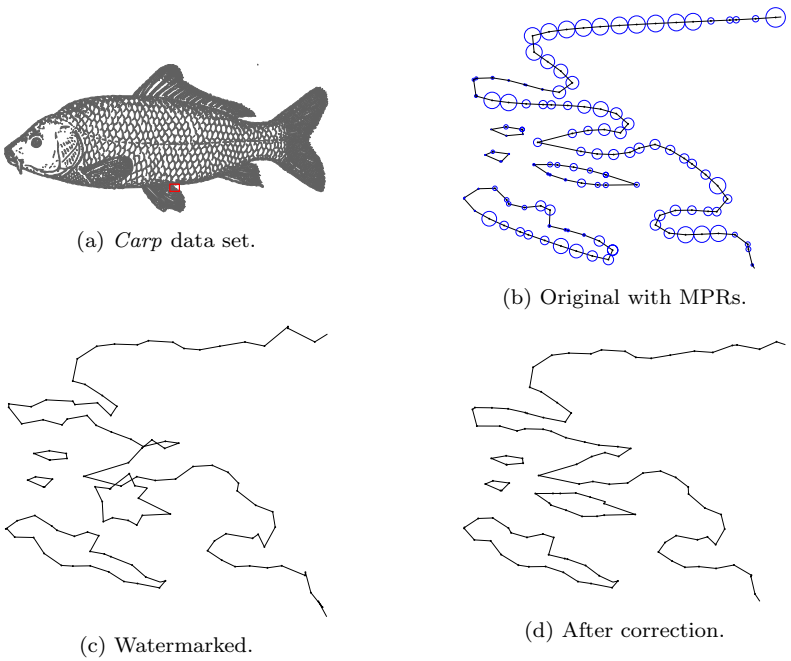


Fig. 11. Probability of Miss (P_m) for the LC detector without and with constraint (MPR and cMPR based on Voronoi and triangulation computation) for varying embedding strength.

The ρ_{LC} detection statistic histograms under $\mathcal{H}_0$ and $\mathcal{H}_1$ (with and without MPR correction) from 1 000 experiments with different, pseudo-random watermarking keys can be observed in Fig. 9 and confirm the assumption of a Normal distribution. Similar results can be obtained for the ρ_{LRT} and ρ_{CD} detection statistic.

In Figs. 12(b) to 12(d) we zoom in on the tail end of the *Carp's* ventral fin. In Fig. 12(b), we show the original vector data with the MPRs superimposed. In Fig. 12(c), the watermarked data is depicted. We can observe that polygonal chains are intersecting (with themselves and other chains). In Fig. 12(d), we show the



(a) *Carp* data set.

(b) Original with MPRs.

(c) Watermarked.

(d) After correction.

Fig. 12. (a) Watermarked *Carp* graphics (24 134 vertices, embedding in 4 890 vertices, and 1 589 vertices MPR corrected). Part of the *Carp* vector data: (b) original data with MPRs superimposed, (c) watermarked without distortion constraint, (d) watermarked after MPR correction.

watermarked data after correction based on the MPRs computed from the original geometry using the Voronoi-based method. Imposing the distortion constraint preserves the geometrical properties, and polygonal chains do not cross.

When MPR correction is applied after the watermarking stage, the correction dampens part of the embedded watermark information. In order to evaluate the detection performance with and without MPR distortion constraint as a function of the embedding strength, we estimate the parameters of the detection statistics ρ under $\mathcal{H}_0$ and $\mathcal{H}_1$ using Monte-Carlo simulation with 1000 test runs. The detection threshold based on $\hat{\mu}_{\rho|\mathcal{H}_0}$ and $\hat{\sigma}_{\rho|\mathcal{H}_0}$ is given by $T_\rho = \sqrt{2}\hat{\sigma}_{\rho|\mathcal{H}_0}\,\mathrm{erfc}^{-1}(2P_{fa}) + \hat{\mu}_{\rho|\mathcal{H}_0}$ for the desired probability of false-alarm (P_{fa}); we set $P_{fa} = 10^{-6}$ for our experiments. Using $\hat{\mu}_{\rho|\mathcal{H}_1}$ and $\hat{\sigma}_{\rho|\mathcal{H}_1}$, the estimated probability for missing the watermark (P_m) can be determined by

$$P_m = \frac{1}{2}\,\mathrm{erfc}\left(\frac{\hat{\mu}_{\rho|\mathcal{H}_1} - T_\rho}{\sqrt{2}\hat{\sigma}_{\rho|\mathcal{H}_1}}\right), \tag{10}$$

assuming the detection statistics follow a Gaussian law, see Barni and Bartolini[2] for details. The probabilities are very low in practice and only rough estimates. They certainly can not be verified directly by experiments, yet they are useful measures to compare different watermarking schemes.

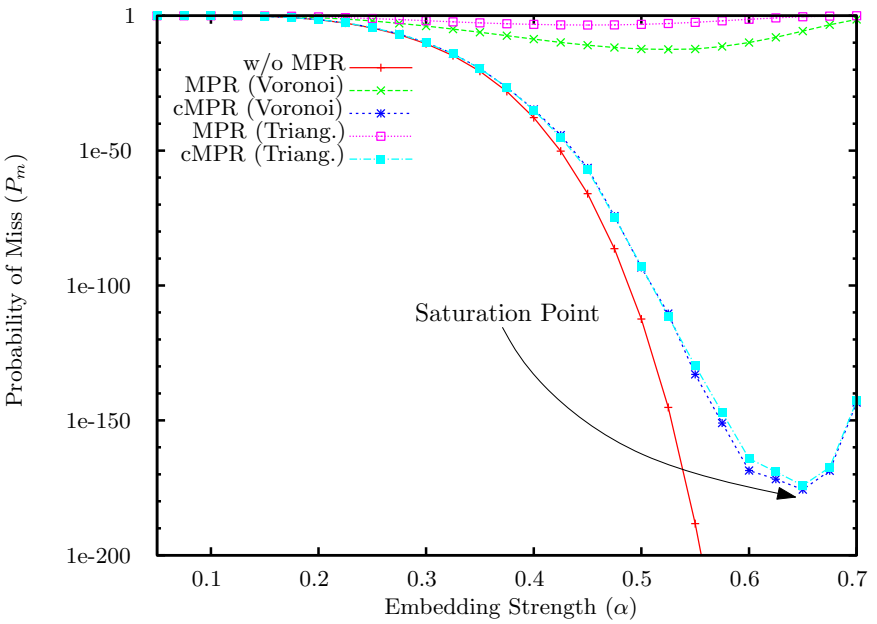


Fig. 13. Probability of Miss (P_m) for the LRT-Rayleigh detector without and with constraint (MPR and cMPR based on Voronoi and triangulation computation) for varying embedding strength.

We plot the probability of missing the watermark with the LC detector based on the *Carp* graphics for a range of embedding strength factors ($\alpha \in \{0.05, 0.075, ..., 0.7\}$) without and with distortion constraint based on MPR and cMPR in Fig. 11. The MPR can be determined with the Voronoi or triangulation-based methods outlined in Secs. 2.1 and 2.2, respectively.

When increasing the embedding strength, more and more watermarked vertices lie outside their MPR and have to be corrected. As expected, this reduces the performance of the watermarking scheme with MPR relative to the unconstrained scheme but guarantees that the geometry of the graphics is preserved. When using the cMPR constraint, performance decreases noticeable only for high embedding strength ($\alpha > 0.5$). Recall that cMPR correction only adjusts watermarked vertices actually causing line segment intersections at the cost of increased algorithmic complexity.

Evidently, the detection performance is worse with the MPRs obtained by the triangulation method. The reason is that the MPRs tend to be smaller, leaving less capacity for the watermark. In the experiment, we used Steiner points to improve the triangulation ($f = 1.5$, c.f. Sec. 2.2). When imposing the cMPR constraint, the difference in detection performance between Voronoi and triangulation is negligible.

The same experiment is conducted with the LRT-Rayleigh detector and results are shown in Fig. 13. Compared to the LC detector, the probability of miss decreases faster for the unconstrained scheme. However, when imposing the MPR

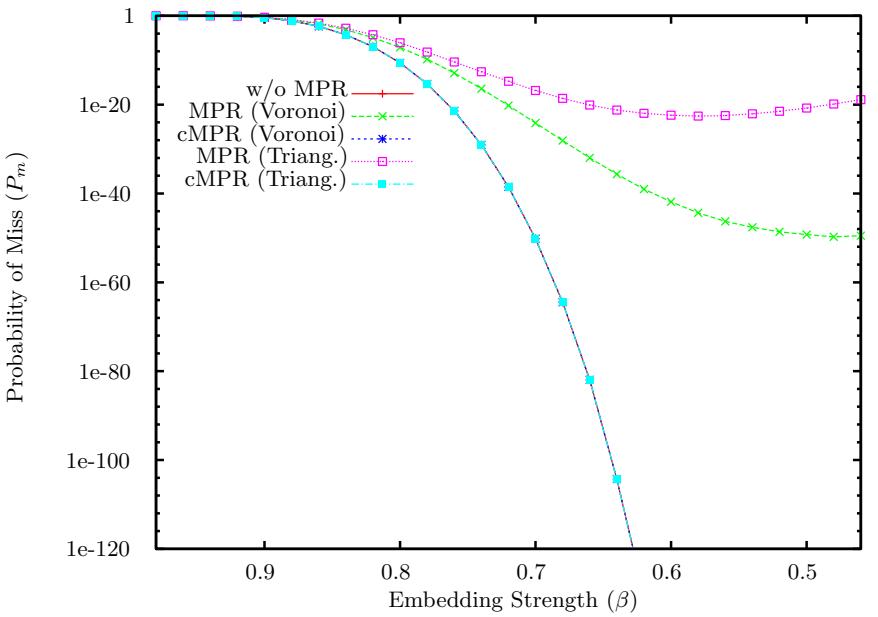


Fig. 14. Probability of Miss (P_m) for the coordinate domain scheme without and with constraint (MPR and cMPR based on Voronoi and triangulation computation) for varying embedding strength. Note that the curves for w/o MPR, cMPR (Voronoi) and cMPR (Triang.) are very close to each other.

constraint we observe that P_m reaches a saturation point around $\alpha = 0.5$. For cMPR, saturation occurs around $\alpha = 0.65$. Remember that the test statistic of the LRT detector depends on knowledge of the embedding strength. In the experiment we match the strength for embedding and detection, an assumption which does not hold when more and more vertices are corrected. Oostveen *et al.*[22] propose a maximum-likelihood estimate of the embedding power for multiplicative watermarking, albeit in a slightly different detection setting.

Figure 14 illustrates the performance of the coordinate domain embedding method[29] with and without MPR and cMPR distortion constraint for the *Carp* data set with varying embedding factors ($\beta \in \{0.98, 0.96, ..., 0.46\}$). When employing the cMPR constraint, the performance of the scheme is identical to the one without MPR as no corrections are necessary for the *Carp* data set (cf. Table 2). This is due to the design of the particular coordinate domain watermarking method aiming at preserving the shape. However, the embedding method can not guarantee that the polygonal chains of the watermarked data set are free of line segment intersections. Table 2 reveals that (few) cMPR corrections are required for certain data sets. Compared with the results of the frequency-domain embedding method (see Table 1), the number of cMPR vertex corrections is considerably reduced. Interestingly, stronger embedding (i.e. smaller β) leads to fewer cMPR corrections in case of coordinate domain watermarking. Remember that strong embedding moves

the non-feature vertices closer to the low-resolution representation of the shape and hence intersections become less likely.

6. Conclusion

Our experiments with both MPR computations – i.e., Voronoi-based and triangulation-based – allow the following conclusions: First, using a Voronoi tessellation yields larger MPR regions and consequently gives more freedom for the watermarking process. Further, it can be extended to more general input, such as circular arcs. Using triangulations, on the other hand, may result in smaller MPRs and less freedom for watermarking. However, in consideration of a potential extension to three dimensions, one might favor this approach due to the simpler implementation.

A potential direction for future work is to consider application scenarios other than copyright protection. While reversibility of the watermarking process is typically not an issue in the literature on copyright protection, it is certainly interesting to investigate potential adjustments to our approach to ensure compatibility with reversible schemes. While a thorough investigation of this issue is out of the scope of this paper, one could envision a strategy to successively reduce the embedding strength of the watermark for instance, until cMPR does not correct any vertices anymore. Whether such an approach can be realized in a principled way is an open problem. Further, it is unclear how detection performance of the watermark would be affected. Investigation of these issues is left for future work.

From a geometry point of view, our work also reveals two interesting problems: How can we compute a triangulation such that the radii of the incircles are maximized? And, more generally, how can we adjust the watermarking phase to respect (as much as possible) other important geometric characteristics of vector data, such as right angles and parallelism?

References

1. I. J. Balaban, An optimal algorithm for finding segments intersections, *Proc. 11th Annu. ACM Symp. Comput. Geom.* (1995), pp. 211–219.
2. M. Barni and F. Bartolini, *Watermarking Systems Engineering* (Marcel Dekker, 2004).
3. M. Barni, F. Bartolini and A. Piva, Improved wavelet-based watermarking through pixel-wise masking, *IEEE Trans. Image Process.* **10**(5) (2001) 783–791.
4. J. Bentley and T. Ottmann, Algorithms for reporting and counting geometric intersections, *IEEE Trans. Comput.* **C-28** (1979) 643–647.
5. M. Bertolotto and M. Zhou, Efficient and consistent line simplification for web mapping, *Int. J. Web Eng. Technol.* **3**(2) (2007) 139–156.
6. A. Bors and M. Luo, Optimized 3D watermarking for minimal surface distortion, *IEEE Trans. Image Process.* **22**(5) (2013).
7. B. Chazelle and L. Palios, Triangulating a nonconvex polytope, *Discr. Comput. Geom.* **5** (1990) 505–526.

8. H.-I. Choi, T.-W. Kim, S.-H. Kwon, H. Moone, S. Park, H.-J. Shin and J.-K. Sohn, Digital watermarking of polygonal meshes with linear operators of scale functions, *Comput. Aided Design* **42**(3) (2010) 163–172.
9. I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich and T. Kalker, *Digital Watermarking and Steganography* (Morgan Kaufmann, 2007).
10. V. Doncel, N. Nikolaidis and I. Pitas, An optimal detector structure for the Fourier descriptor domain watermarking of 2D vector graphics, *IEEE Trans. Visualizat. Comput. Graph.* **13**(5) (2007) 851–863.
11. D. Douglas and T. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, *Can. Cartographer* **10**(2) (1973) 112–122.
12. R. Estkowski and J. Mitchell, Simplifying a polygonal subdivision while keeping it simple, *Proc.* 17*th Annu. ACM Symp. Comput. Geom.* (2001), pp. 40–49.
13. J. Fridrich, M. Goljan and R. Du, Lossless data embedding – New paradigm in digital watermarking, *EURASIP J. Appl. Signal Process.* **2002**(2) (2002) 185–196.
14. I. Guskov, K. Vidimce, W. Sweldens and P. Schroder, Normal meshes, *Proc. ACM SIGGRAPH, '*00 (2000), pp. 95–102.
15. M. Held, VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments, *Comput. Geom. Theor. Appl.* **18**(2) (2001) 95–123.
16. M. Held, VRONI and ArcVRONI: Software for and applications of Voronoi diagrams in science and engineering, *Proc.* 8*th Int. Symp. Voronoi Diagrams in Science & Engineering* (2011), pp. 3–12.
17. M. Held and S. Huber, Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight-line segments, *Comput. Aided Design* **41**(5) (2009) 327–338.
18. S. M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*, Vol. 2 (Prentice-Hall, 1998).
19. A. Khoshgozaran, A. Khodaei, M. Sharifzadeh and C. Shahabi, A hybrid aggregation and compression technique for road network databases, *Knowl. Infor. Syst.* **17**(3) (2008) 265–286.
20. T. Lambert, The Delaunay triangulation maximizes the mean inradius, *Proc.* 6*th Canadian Conf. Comput. Geom.* (*CCCG'*94) (1994), pp. 201–206.
21. R. Ohbuchi, H. Ueda and S. Endoh, Watermarking 2D vector maps in the mesh-spectral domain, *Proc. Int. Conf. Shape Modeling and Applications* (*SMI'*03) (2003).
22. J. Oostveen, T. Kalker and J.-P. Linnartz, Optimal detection of multiplicative watermarks, *Proc.* 10*th Europ. Signal Proc. Conf.* (*EUSIPCO '*00), Vol. 5 (2000), pp. 2973–2976.
23. J. Pan, J. Zheng and G. Zhao, Blind watermarking of nurbs curves and surfaces, *Comput. Aided Design* **45**(2) (2013) 144–153.
24. C. I. Podilchuk and W. Zeng, Image-adaptive watermarking using visual models, *IEEE J. Selected Areas in Commun. Special Issue on Copyright and Privacy Protection* **16**(4) (1998) 525–539.
25. S. Popinet, GTS – The GNU triangulated surface library (2006), online available from http://gts.sourceforge.net/.
26. Y.-C. Pu, W.-C. Du and I.-C. Jou, Perceptually transparent polyline watermarking based on normal multi-resolution representation, *IEICE Trans. Infor. Syst.* **E89-D**(12) (2006) 2939–2949.

27. U. Ramer, An iterative procedure for the polygonal approximation of plane curves, *CGIP* **1**(3) (1972) 244–256.
28. A. Saalfeld, Topologically consistent line simplification with the Douglas-Peucker Algorithm, *Cartogr. Geogr. Infor. Sci.* **26**(1) (1999) 7–18.
29. C. Y. Shao, H. L. Wang, X. M. Niu and X. T. Wang, Shape-preserving algorithm for watermarking 2-D vector map data, *Proc. 7th IEEE Workshop Multimedia Signal Proc.* (2005), pp. 1–4.
30. J. Shewchuk, Delaunay refinement algorithms for triangular mesh generation, *Comput. Geom. Theor. Appl.* **22**(1–3) (2002) 21–74.
31. V. Solachidis and I. Pitas, Watermarking polygonal lines using Fourier descriptors, *IEEE Comput. Graph. Appl.* **24**(3) (2004) 44–51.
32. M. Voigt, B. Yang and C. Busch, Reversible watermarking of 2D-vector data, *Proc. 2004 Workshop on Multimedia and Security, MM&Sec'04*, Magdeburg, Germany (ACM, 2004), pp. 160–165.
33. A. B. Watson, H. A. Solomon, A. Ahumada and A. Gale, DCT basis function visibility: Effects of viewing distance and contrast masking, ed. B. Rogowitz, *Human Vision, Visual Processing, and Digital Display IV*, Vol. 2179 (1994), pp. 99–108.