Geographical Database Watermarking by Polygon Elongation (Technical Report)

Julien Lafaye[1], Jean Béguec[1,3], David Gross-Amblard[1,2] and Anne Ruas[3]

[1] Laboratoire CEDRIC, Spécialité Informatique – CC 432, Conservatoire national des arts & métiers, 292 rue Saint Martin, 75141 PARIS Cedex 3, France
[2] Laboratoire LE2I, Université de Bourgogne, Faculté des Sciences Mirande, Aile de l'ingénieur, BP 47870 21078 DIJON Cedex, France
[3] Laboratoire COGIT, Institut Géographique National (IGN), 2/4 Avenue Pasteur 94 165 SAINT MANDE Cedex, France

Abstract. Due to the ease of digital copy, watermarking is crucial to protect the intellectual property of rights owners. We propose an effective watermarking method for vectorial geographical databases, with the focus on the buildings layer. Embedded watermarks survive common geographical filters, including the essential squaring and simplification transformations, as well as deliberate removal attempts, e.g. by noise addition, cropping or over-watermarking. The impact on the quality of the datasets, defined as a composition of point accuracy and angular quality, is assessed through an extensive series of experiments. Our method is based on a quantization of the distance between the centroid of the building and its extremal vertex according to its orientation.

1 Introduction

Geographical Information Systems (GIS) have existed for more than 40 years but their application domain is much wider nowadays, ranging from environmental surveillance by country agencies to localization-aware services for individual mobile users. This phenomenon is stressed for the general public by the increasing availability of GPS devices (e.g. car navigation) and the recent development of Google Earth and GeoPortail [1]. Most geographical applications rely on an underlying vectorial spatial database (points, polylines and polygons). Even in Google Earth or GeoPortail, where the user interface is image-oriented, the provided satellite images are semantically underlined with polygonal structures representing points of interest (viewpoint, services), buildings or road networks.

Gathering such accurate information is an onerous task for the data owner. Then, huge and detailed vectorial databases carry a high scientific and/or economical value. For example, 50 USD is the smallest fee to be licensed to use polygons from a narrow one square kilometer area. This price is 10 times higher for a reproduction license and far much higher for a full commercial license. Due to the ease of reproduction of digital media, unauthorized copy and use threaten geographical data providers. Protecting the intellectual property (IP) of rights owner is a requirement.

On the legal side, data providers restrict the way buyers are allowed to use their data. On the technical side, robust watermarking is a known technique for IP protection. It consists of hiding a copyright mark within the document. Embedded marks must be robust against removal attempts to be effective. In this paper, we propose a robust watermarking method for polygonal datasets.

To embed the watermark, the data has to be altered. What might sounds as a drawback is common to most watermarking methods [14]. There is a trade-off between watermark robustness and data alteration: the more alterations are allowed, the more robust the embedded watermark is. So, defining precisely what makes the value of a dataset is a prerequisite for watermarking.

Some applications do not rely only on spatial accuracy (i.e. the distance between a point in the real world and this point in the dataset). For example, spatial accuracy is not crucial for tourist city maps designers who apply strong transformations to road polylines and building polygons in order to increase legibility. Some others focus on objects like forests, cliffs and shallows for which precise borders can be somewhat difficult to define. But many applications rely on accurate data for automatic operations (e.g. service proximity search, GPS navigation, spatial analysis of risks, etc.). Accuracy can even be mandatory, e.g. for reefs locations on IHO/SHOM boat maps [27]. Finally, accurate datasets must conform with some standard reference system for interoperability purposes (e.g. the World Geodetic System – WGS84, which is the GPS reference system).

Real world requirements entail specific constraints within the dataset, e.g. right angles between walls of buildings. Because of acquisition errors and artifacts, digital geographical data needs to be corrected so that it reflects these constraints. The application of these corrections is considered as an essential step to optimize the quality of the dataset. It turns out that most of the vectorial content of geographical databases consists of building polygons (80% on the professional dataset used in the experiments), which constitute the primary focus of our work. For this kind of data, the *squaring* transformation is used. It consists of moving the vertices of the dataset so that almost right angles become exact right angles. Experiments show that it also tends to increase accuracy. In this paper, we model the quality of a dataset by mean of its (1) *accuracy* and (2) its *angular energy*. As pointed out in a recent survey [19], quality (fidelity in [19]) must take into account accuracy but also other parameters including the regularity of the shapes contained within the dataset. Here, this is achieved by considering angular energy.

Considering the specificities of the input data is mandatory to achieve robustness of a watermarking method. Indeed, methods introducing noise removed by the corrections, including squaring, are of no use if users systematically apply them. Moreover, the huge volume of these datasets and their constant updates motivate the use of blind watermarking algorithms, i.e. methods that do not require the original dataset to perform watermark detection. There exist several recent techniques for databases watermarking. Some of them apply to relational databases [3], others to geographical datasets [19, 22, 25]. None of them takes into account the squaring transformation.

It this paper, we propose an effective method for building watermarking that is robust against geographical transformations (including squaring and simplification) and attacks by malicious users. As far as we know, this is the first method which takes into account the essential squaring transformation. It provides a high level of security while controlling the impact on the quality of the dataset (point accuracy and angular quality) and not introducing topological errors (overlapping polygons). The scheme is blind: the original dataset is not required for detection. We do not assume that primary keys identifying polygons are available. Finally, it is incremental and its complexity is linear in the size of the dataset, enabling scalability.

A classical skeleton [3] of databases watermarking algorithms is to create a secret dependency between (1) a robust identifier of the data and (2) one of its characteristics, e.g. between the primary key of a tuple and one of its numerical attributes. Revealing this dependency acts as a proof of ownership. In our approach, we get rid of the primary key by constructing a robust identifier for each building using well chosen high significant bits of its centroid. Then, we rely on the observation that buildings have an intrinsic orientation and that most of their edges are parallel or perpendicular to this orientation. To hide a watermark bit, we expand or shrink buildings along their orientation. The expansion ratio is deterministically chosen among a set of quantized values according to the robust identifier of the polygon, the secret key of the owner and the bit to be embedded. By embedding the watermark within the shapes of a building rather that within the coordinates of its vertices, we achieve robustness of watermarks against squaring. Our scheme is also robust against other transformations we present later in the paper. Any naïve user or malicious attacker has to tremendously reduce accuracy and/or angular quality of the dataset to erase the watermark.

Outline After a description of watermarking basics, a simple model for buildings databases and a definition for data quality are presented in Section 2. Our watermarking procedure is described in Section 3. Correction, efficiency and robustness of the method are assessed in Section 4, through an extensive series of experiments. Related work is exposed in Section 5 and Section 6 concludes.

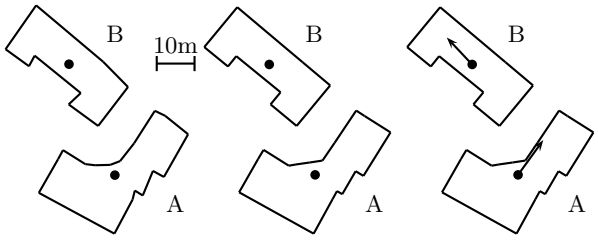
2 Preliminaries

2.1 Quality of Geographical Data

We suppose, as in any geographical application, that a reference system R_0 has been chosen and that all spatial coordinates are expressed in R_0 (e.g. cartesian coordinates on the World Geodetic System, or WGS84).

A point $p = (x, y)$ is defined by its 2-dimension coordinates (x, y) in some reference system R_0. A simple polygon $P = (p_1, \ldots, p_n, p_{n+1} = p_1)$ is described by the list of its points. Two polygons taken from a real dataset are shown on Fig. 1(a). A geographical database instance is defined by (R, DB) where R is a reference system and $DB = \{P_i\}$, $i \in \{1, \ldots, N\}$ is a set of N polygons.

It is always provided with some reference system otherwise it is of no use for automatic operations (nevertheless, we discuss in Section 4.4 the problem of missing reference system).



(a) Raw data (b) Squared data (c) Orientation

Fig. 1. Buildings polygons

We do not rely on the order of polygons within the dataset, nor on the order of points within a polygon. Furthermore, there is no primary key identifying these polygons. Polygons are supposed non-overlapping as in many geographical applications. They can have holes. In that case, we process them as the full polygon having the same envelope. More sophisticated models of spatial data expressing topology exist, but we omit these enhancements for the sake of simplicity.

The (economical) value of a dataset (R, DB) is correlated with its *mean accuracy*, its *maximum accuracy* and its *angular quality*. The *mean accuracy* mean value of the distance between a point of a building and its corresponding point in the dataset; the *maximum accuracy* is the maximum value of the distance between a point of a building and its corresponding point in the dataset. The *angular quality* [4] is defined as the opposite of its angular energy. The energy of an angle is a continuous piecewise quadratic whose minima are reached for multiples of $\pi/4$. The angular energy of a polygon is the sum of the energies of its angles. The intuition is that angles of real-world buildings are mostly right, or at least multiples of $\pi/4$. So, regular buildings have low energy levels.

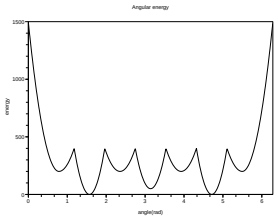


Fig. 2. Minima of Angular Energy ($k\pi/2$)

Observe that printing/scanning a map drawn from the dataset, translating, applying an affine transformation like rotating the whole dataset are not an issue, since we focus on datasets with a reference system.

2.2 Watermarking

A watermarking procedure is defined as a pair of algorithms $(\mathcal{W}, \mathcal{D})$, where $\mathcal{W}$ is the watermarking algorithm, and $\mathcal{D}$ is the detection algorithm. Algorithm $\mathcal{W}$ takes as inputs a dataset (R, DB) , a secret key $\mathcal{K}$ and some tuning parameters, and produces a watermarked dataset $(R, \mathcal{DB}_{\mathcal{K}})$. The aim of the detector is, given a suspect dataset (R', DB') and the secret key $\mathcal{K}$, to decide whether this dataset holds a watermark or not. A watermarking procedure is said to be *blind* if the original dataset is not needed by the detector $\mathcal{D}$. It is said to be *robust* if it detects marks in altered watermarked datasets. It is well known that any robust watermarking method must alter the data. Hence, there is a trade-off between the allowed alteration, i.e. the allowed impact on the quality, and the robustness of the algorithm.

To evade detection, an attacker may use one of the following attacks: random alteration of point positions, mixing polygons from various datasets, applying the same or another watermarking algorithm, and specific attacks like polygon-wise rotations. We discuss these attacks in Section 4. Of course, the attacker, who still wants to re-sell a valuable dataset must adopt a common reference system and limit the quality loss so that profit can still be made from the attacked database.

2.3 Geographical Filters & Attacks

Geographical datasets are likely to undergo transformations by legitimate or malicious users. A broad collection of such transformations is presented below. They can be divided into correction filters (SQ, DP), legibility improvements (ETR, MBR, CE) and malicious attacks (GN, OW, CA). Nevertheless, this taxonomy is not fixed as a malicious user might apply correction filters and a legitimate one might crop a large dataset to keep only the part useful to him. A robust enough watermarking algorithm should resist all of them:

Squaring (SQ) For each polygon, its vertices are moved so that its angular energy is lowered. The strength of the squaring is controlled by the maximum allowed alteration d on coordinates. Fig. 7(a) shows a squared building.

Douglas-Peucker simplification (DP) The Douglas-Peucker simplification algorithm [6] is a polyline simplification algorithm. It works by removing the vertices of polygons that draw small artifacts on the edges of this polygon. Its strength is controlled by a threshold distance d. The higher d, the larger the removed artifacts are. See Fig. 7(c) for an example of the application of a DP filtering with $d = 5m$.

Cropping (CA) Polygons not contained within a given rectangularly shaped region of the dataset are discarded.

Gaussian noise (GN) A random noise is added to each point of the database. The distribution of the noise has mean 0 and a variable deviation d.

Over-watermarking (OW) Applying the watermarking algorithm with a different key on an already marked dataset.

Enlarge to rectangle (ETR) This filter replaces buildings by their bounding rectangle. Two modes are available. The first one replaces each building with a rectangle with the same surface. The second one takes as input a target scale and replaces the buildings that are two small (for a legally fixed threshold value) to be legible on a map at that scale.

Change elongation (CE) Applies a fixed factor elongation along their orientation on all buildings of the dataset.

Minimum bounding rectangle (MBR) Replaces each building by its minimal bounding rectangle.

3 Building Watermarking

3.1 Outline of the Algorithm

The rationale for many watermarking algorithms is to hide a secret dependency between (1) a robust part of the dataset, that will survive most alterations, and (2) one of its characteristics, whose alteration is allowed up to a reasonable limit. Revealing this secret dependency acts as a proof of ownership. We build a robust identifier id_i for each polygon P_i by using the highest significant bits of the coordinates of its centroid, expressed in the predefined reference system R_0. This identifier is robust since it is invariant through the modifications of vertex coordinates, involving only least significant bits. High amplitude modifications are likely to break the identifiers but also to lead to visible shapes alterations and/or polygon overlappings. Furthermore, if the coordinates of the polygon of the centroid are expressed in a reference system R', different from R_0, it is easy to convert them back into R_0. Indeed, no geographical data comes without a reference system.

In order to hide a bit of information in polygon P_i, we expand or shrink it(i.e. expand with a ratio < 1) along its orientation. This orientation is computed relatively to the centroid (see Fig. 1(c)), and represents the majority weighted angle among edges directions. We present its computation in Section 3.3. For a rectangular shape, this orientation is parallel to the longest edge. Choosing to expand along this orientation offers several advantages. First, we observed that most edges of a polygon are parallel or perpendicular to this orientation. For example, there are 3 directions in polygon A (Fig. 1(b)): SW-NE, SE-NW and W-E. The main direction, i.e. the orientation is clearly SW-NE since the longest edges are heading this direction. Other directions are perpendicular or make a $\pi/4$ angle with the orientation. When a polygon is expanded along its orientation, geometrical relations between directions do not change. Second, an expansion along the orientation can still be detected if the polygon is rotated. Finally, the impact on polygon surfaces has tighter bounds compared to the case where shrinking or expanding along several directions is allowed.

It remains to compute the expansion factor to apply, and to choose which polygons are going to be altered. These operations must be done so that any attacker, aware of the watermarking method, is unable to guess on which polygons they were actually applied. A classical method to achieve this [3] is the following: use the concatenation of the given identifier id_i of a polygon and the secret key $\mathcal{K}$ of the owner to seed a secure pseudo-random number generator (PRNG). Use pseudo-random drawings from the generator to determine whether the current polygon is modified and, eventually, with which expansion factor. The sequence of numbers produced by the generator is predictable if and only if $id_i.\mathcal{K}$ is known. It appears purely random to anyone who does not possess this seed (an attacker may easily compute id_i, but $\mathcal{K}$ remains unknown).

In the following, we detail the three consecutive steps of our algorithm: (1) Computation of polygon identifiers and orientations, (2) Computation of expansion factors and (3) Watermarking by expansion.

Example 1. An example of our watermarking method applied on polygons A and B is shown on Fig. 3. Original shapes are shown in black and watermarked ones in grey. First, we compute the centroid of A and B, obtaining $O_A = (293, 155)$ and $O_B = (171, 447)$. To form unique identifiers id_A and id_B, we concatenate the two high significant digits of each coordinate, obtaining $id_A = 2915$ and $id_B = 1744$. Choosing these two digits is correct under the hypothesis that any reasonable alteration is below 10 meters and that the typical distance between any two buildings is more than 10 meters (this example considers decimal base while our algorithm considers binary base). Second, based on the pseudo-random choices of a generator seeded with id_A and the secret key $\mathcal{K}$, we decide that A must be watermarked with a mark bit 0. We compute the main orientation $\boldsymbol{u}$ of A and find the vertex p such that $\boldsymbol{u}.Op$ is maximal. Let x_{max} denote this value. Finally, we expand the building along its main orientation so that x_{max} becomes a predefined value x^0_{max}, encoding bit 0. Polygon B is processed identically. Remark that A has been expanded whereas polygon B has been shrinked, and that most angles are invariant under this transformation.

3.2 Computing Robust Identifiers

As a robust identifier, we use the highest significant bits of the centroid of the polygon. If P is a polygon with n points $p_1, \ldots, p_n, p_{n+1} = p_1$, its area A and its centroid $O = (x_0, y_0)$ can be computed using the following formulae [5]:

$$A = \frac{1}{2} \sum_{1 \leq i \leq n} (x_i y_{i+1} - x_{i+1} y_i),$$

$$x_O = \frac{1}{6A} \sum_{1 \leq i \leq n} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i),$$

$$y_O = \frac{1}{6A} \sum_{1 \leq i \leq n} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i).$$

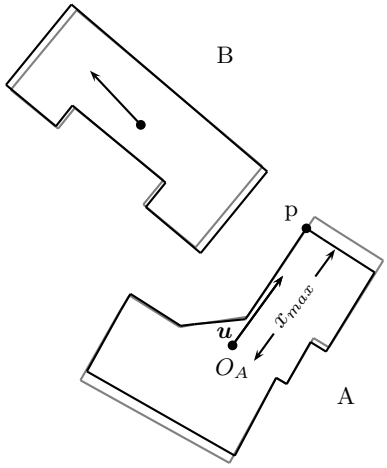


Fig. 3. Bit Embedding by expansion

Centroids of polygons A and B are represented as black dots on Fig. 1(a) and 1(b).

We need to ensure that the chosen highest significant bits are significant enough. Suppose that the h-th bit is the least highest significant bit. On the one hand, h must be high enough to that small modifications of the polygon do not change it. On the other hand, h must be small enough so that two adjacent polygons do not share the same identifier. As we will see later, two watermarked polygons sharing an identifier will be expanded using the same quantization step. Such an information may be used by an attacker to break the watermark. Therefore, shared identifiers must be avoided. A good choice is to select the smallest value h for which the identifiers vary from one polygon to the other. We illustrate how we compute h by means of an example. For our real-life dataset (detailed in Section 4), we measured, for each polygon, the distances between the centroid and the farthest vertex from the centroid. We obtain a mean of 12 meters and a standard deviation of 10.5 meters, that is, most polygons have an interior maximum distance between $2(12-10.5) = 3$ meters and $2(12+10.5) = 50$ meters. Considering 3 meters as a minimum size for a polygon, most polygons have their centroid spaced by at least 6 meters. Even if there are pathological configuration, it is very likely that, by choosing $h = 2$, two polygons do not share the same identifier. When $h = 2$, $l = 2^h = 2^2 = 4$m is considered as the minimum significant distance. Experimentally, we verified that, by choosing $h = 0, 2, 4$ and 8, we had $0, 0, 40$ and 300 cases of identifiers collision out of 4278 polygons. Hence, it seems that a good heuristic is to use $h = \lfloor \log_2(2.(\bar{l} + \delta l)) \rfloor$ if $\bar{l}$ and δl are the mean and the deviation of the main lengths of the polygons in the dataset.

The identifier of a polygon P is computed by pruning in the binary representations of its x and y coordinates the bits that represent powers of two at most

$h-1$ and concatenating them. We denote by $hsb(O,h)$ this operation.

$$id = hsb(O,h) = concat(hsb(x_O,h), hsb(y_O,h)).$$

3.3 Computing Polygon Orientation

We define the main orientation $\boldsymbol{u}$ of a polygon as the maximum weighted orientation of its edges. For instance, if only e_1 and e_2 have orientation α, then the weight of angle α is the sum of the lengths of e_1 and e_2. The problem is that parallel walls in the real world are not necessarily mapped to parallel edges in the dataset. So, we need to sum the lengths of edges that are almost parallel. We define ε the tolerance angle, that is e_1 and e_2 are considered as having the same orientation if their orientations α_1 and α_2 are such that $|\alpha_1 - \alpha_2| < \varepsilon$. To efficiently compute the orientation, we defined a bucket-based classifying algorithm based on the observation that there is often only a small number of different orientations per polygon. The algorithm consists of the following three steps. First, we create a set of k empty buckets, provided we choose k such that $\pi/k < \varepsilon$. In bucket i, we put all edges having an orientation between $(i-1).\frac{\pi}{k}$ and $i.\frac{\pi}{k}$. Hence, in buckets i and $i+1$ we have all edges that are almost equal to $i.\pi/k$. Then, we aggregate these small buckets into bigger ones by merging two buckets if there is no empty bucket between them. The main orientation of a building is computed as the mean value of the bucket having the highest cost (the cost of a bucket being defined as the sum of the lengths of the edges in that bucket). It can happen that three or more buckets need to be aggregated, leading to consider orientations as equal when their difference is greater than angle tolerance. This is very unlikely. Indeed, we observed that on buildings, there is only a few directions per polygon (2, 3 in most cases) which are clearly separated. A similar approach was followed in [7] with the main difference that the method proposed in [7] requires to compute the weight of all π/k orientations and select the one with the highest weight. Our method is more efficient but may be more accurate in a restricted number of situations.

Example 2. We illustrate the orientation computation algorithm on polygon A of Fig. 1(b). The number of classes is set up to 10. Table 1 contains the number of edges and the weight of each bucket. The highest cost bucket is bucket 3, i.e. the orientation is between $3\pi/10$ and $4\pi/10 = 2\pi/5$. The computation of the weighted mean angle of bucket 3 gives 0.96 rad.

Table 1. Angles Buckets

bucket	0	1	2	3	4	5	6	7	8	9
#edges	3	0	0	8	0	0	0	0	6	0
weight	9.02	0	0	62.4	0	0	0	0	45.2	0

For some specific shapes, e.g. perfect squares and circles, our definition of orientation is ambiguous. Instead of arbitrarily choosing an orientation, we simple ignore these unusual cases for polygon expansion. On the test sample of our experiments, we had to ignore 1 polygon out of 4278.

3.4 Expansion as a Bit Embedding Method

In this subsection we show how to embed a single watermark bit b into a polygon P. To ensure that the watermark is robust enough, we alter the overall shape of the polygon. More precisely, we alter the longest distance x_{max} along the orientation $\boldsymbol{u}$ from the centroid O to a vertex p. For a rectangular polygon, this length is half the length of the longest edge. We name *main length* this longest distance. But only altering the coordinates of p is not sufficient because it may lower angular quality (right angles may be flattened by this transformation). Hence, we choose to alter all lengths along the orientation $\boldsymbol{u}$ so that most angles are preserved. Defining by $\boldsymbol{v}$ the unary vector such that $(0, \boldsymbol{u}, \boldsymbol{v})$ is a direct orthonormal basis, watermarking is done as follows:

- compute the x coordinate x_i of each point p_i of the polygon in $(0, \boldsymbol{u}, \boldsymbol{v})$;
- compute the main length $x_{max} = \max_i\{|x_i|\}$;
- expand all points coordinates along direction $\boldsymbol{u}$ so that x_{max} becomes one of the values $\{x^0_{max}, x^1_{max}\}$ coding a watermark bit 0 or 1. This later operation on x_{max} is known as quantization (see below).

This paragraph details quantization. Given a quantization step d, we define 0-quantizers (resp. 1-quantizers) as $q^k_0 = k.d$ (resp. $q^k_1 = k.d + d/2$), $k \in \mathbb{Z}$. Intuitively, 0-quantizers (resp. 1-quantizers) are used to code a bit 0 (resp. a bit 1). To quantize the value x_{max} using the i-quantizers ($i \in \{0, 1\}$), we look for k_0 such that $|q^i_{k_0} - x_{max}|$ is minimal. More precisely, this is achieved with the following steps (quantization on 0-quantizers is presented):

- compute $k_r = x_{max}/d$;
- round k_r to the closest integer k_0;
- define the quantized version of x_{max} as $x'_{max} = k_0.d$.

The quantization process is illustrated on Fig. 4.

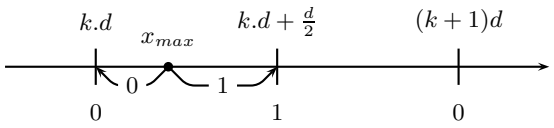


Fig. 4. Encoding 0 or 1 into the main length x_{max} using quantization

The expansion coefficient of the polygon is defined as $\sigma = {x'}_{max}/x_{max}$. We transform each point $p = x.\boldsymbol{u} + y.\boldsymbol{v}$ in the original polygon into a point $p' = \sigma.x.\boldsymbol{u} + y.\boldsymbol{v}$ in the watermarked polygon.

$$p = x.\boldsymbol{u} + y.\boldsymbol{v} \quad \text{original point } p$$
$$p' = \sigma.x.\boldsymbol{u} + y.\boldsymbol{v} \quad \text{modified point } p'$$

The expansion is such that the maximum distortion on a vertex of a polygon is at most $d/2$. Remark that this distortion can be reached only for the vertices that are the farthest to the centroid along $\boldsymbol{u}$. On the average, and for these points, the actual distortion is $d/4$.

3.5 Watermarking Algorithm

Watermarking The complete algorithm is presented in Alg. 1. Let $1/\gamma$ be the target ratio of watermarked polygons. It is a parameter of the algorithm. For each polygon of the dataset, we compute its robust identifier *id*. Then, we seed a pseudo-random number generator G (PRNG) with $\mathcal{K}.id$. If the first integer produced by G modulo γ is 0, we embed a bit in the polygon. The bit is chosen according to the next binary value produced by G and embedded using the previously described expansion method.

Variable Step Quantization We do not use a single quantization step d but a quantization interval $[d_{min}, d_{max}]$. Indeed, if d is the same for the whole dataset, main lengths of all watermarked polygons will be multiples of d. This could be easily detected and used by an attacker to alter the watermark [16]. To prove this, we measured the cumulative distributions of main lengths in three datasets: an unwatermarked one, a watermarked one using a fixed quantization step and another watermarked one using a variable quantization step, randomly chosen. For the two watermarked datasets, γ was set to 1 to emphasize the differences. Graphs are shown on Fig. 5. We notice that, when the quantization step is fixed, the cumulative distribution is piecewise constant whereas it is very close to the original one when a variable quantization step is used. Fixed-step quantization is too visible through statistical analysis. So, we use a different quantization step for each watermarked polygon.

Discussion Using this method, the watermark is spread almost uniformly over the dataset. This process being controlled by a secret key, it is impossible to find the exact locations of polygon expansions, assuming PRNG are secure. To alter the watermark, an attacker has to alter much more polygons than the watermarking process did if he wants to be sure to affect all watermarked polygons. The choice of watermarking parameters γ, d_{min} and d_{max} depends on the specific usage of the dataset. They cannot be fixed arbitrarily for all applications but the following rules are always valid:

- there is an unavoidable trade-off between quality alteration ($\gamma \uparrow, d_{min} \downarrow, d_{max} \downarrow$) and robustness of the watermark ($\gamma \downarrow, d_{min} \uparrow, d_{max} \uparrow$). Experiments presented in Section 4 give indications on how to choose optimal values;

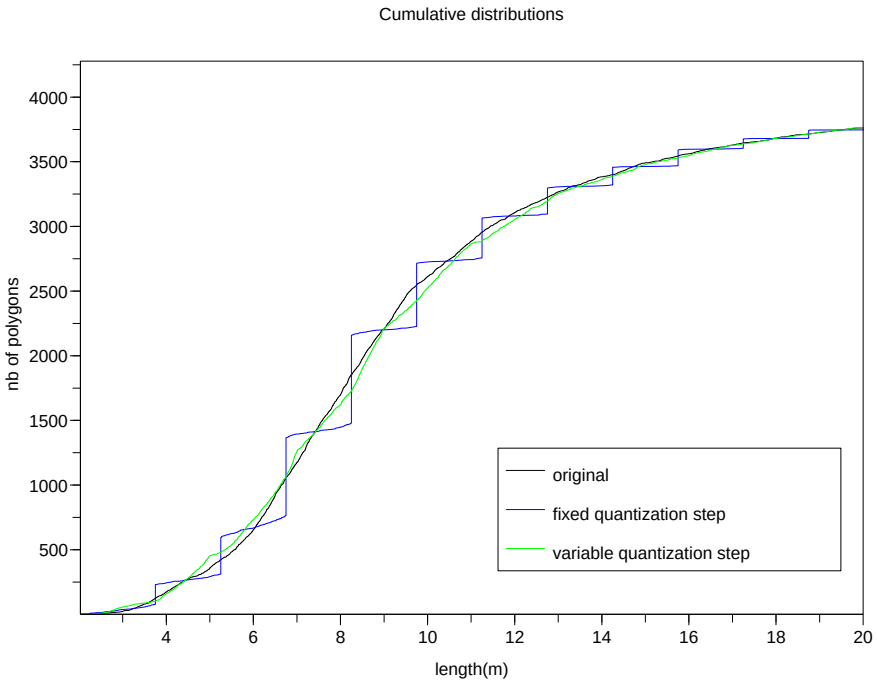


Fig. 5. Statistical invisibility of watermarked data using variable step quantization

– if the accuracy (maximum distance between a point in the dataset and in the real world) of the unwatermarked database is β_1, then the accuracy of the watermarked one is $\beta_1 + d_{max}/2$. If the watermarked dataset is sold under the agreement of an accuracy β_2, then d_{max} must be chosen so that $d_{max} < 2(\beta_2 - \beta_1)$;
– the allowed alteration on the building, i.e. d_{max} must be higher than the typing accuracy of the dataset. Below this value, alterations can be considered as noise and rounded by a malicious user without altering the quality of the dataset at all. For instance, a 1mm alteration is meaningless in a dataset of accuracy 1m.

3.6 Handling Data Constraints

The bit embedding method using expansion does not take into account topological relationships between buildings. We voluntarily chose to ignore them during bit embedding and to detect errors and cancel modifications when needed (function `testCollisions`). Such a strategy is valid as soon as a few errors occurs. By choosing $d_{max} = 4$ meters, the alteration on each point of a polygon is at most 2 meters. Usually, even in urban areas, polygons are spaced by a distance superior to 2 meters. Indeed, with this value, we got only one case of overlapping, even in the worst setting, i.e. when $\gamma = 1$. This validates the detect-and-cancel strategy. Such a post-watermarking filtering enables to handle any kind of errors which can occur sparsely during the watermarking process. Observe that since bit embedding is a local operation, collisions can only occur is a small area around the polygon. This allows the use of classical spatial indexing techniques to check for collisions.

3.7 Detection

Outline Given a suspect dataset (R', DB'), we first translate it into the original reference system R_0, obtaining (R_0, DB'). The detection algorithm is very similar to the watermarking algorithm, with the essential difference that no alteration is performed. It consists of two steps: computing the ratio of matching polygons and comparing this ratio to a predefined threshold value α. The values of $d_{min}, d_{max}, h, \gamma$ and $\mathcal{K}$ used for detection must be the same as the ones used for watermarking. So they must be kept as part of the secret. For each of the polygon we seed a random generator with $\mathcal{K}$ concatenated with its identifier. If the polygon satisfies the watermarking condition (i.e. $\texttt{nextInteger}(G) \bmod \gamma = 0$), we compute the expected bit value b as $\texttt{nextInteger}(G) \bmod 2$. We also compute the quantization step d between d_{min} and d_{max}. Then, we decode the bit b' embedded in the main length x_{max} of the polygon and compare it with b.

Decoding To decode a bit from a quantized value x, we simply check whether it is one of the 1-quantizers or one of the 0-quantizers. If x is none of the i-quantizers, we compute the closest quantized value x'_1 in 1-quantizers and the

Algorithm 1: Watermarking algorithm

Input: secret key $\mathcal{K}$, watermarking ratio $1/\gamma$, h,quantization step interval $D = [d_{min}, d_{max}]$
Data: (R_0, DB): original dataset
Output: $(R_0, DB_{\mathcal{K}})$: watermarked dataset
foreach *building* P *in* DB **do**
  $O \leftarrow$ `centroid`(P);
  $id \leftarrow$ `hsb`(O, h); `/* robust identifier` id `*/`
  `seed`$(G, \mathcal{K} \cdot id)$; `/* seed the PRNG` G `with` $\mathcal{K} \cdot id$ `*/`
  if `nextInteger`(G) $\mod \gamma = 0$ **then**
    `// Watermark this building`
    $\boldsymbol{u} \leftarrow$ `orientation`(P); `/* orientation */`
    $x_{max} \leftarrow \max\{p \in P | \mathbf{Op} \cdot \boldsymbol{u}\}$; `/* main length */`
    $d \leftarrow d_{min} +$ `nextFloat`$(G).(d_{max} - d_{min})$; `/* quantization step */`
    $b \leftarrow$ `nextInteger`(G) $\mod 2$; `/* watermark bit` b `*/`
    $x'_{max} \leftarrow$ `quantize` (x_{max}, d, b); `/* quantize` x_{max} `*/`
    $\sigma \leftarrow x'_{max}/x_{max}$; `/* expansion factor */`
    `expand`$(P, O, \boldsymbol{u}, \sigma)$;
    if `testCollision` $()$ **then**
      `rollback` () ;

closest quantized value x'_0 in 0-quantizers. We compare the distance $d_0 = |x'_0 - x|$ and $d_1 = |x'_1 - x|$. If $d_0 < d_1$, we decode a bit 0; if $d_0 > d_1$, we decode a bit 1. If $d_0 = d_1$ no bit can be decoded. Note that a quantized value, with step d, can be altered up to $d/4$ without leading to a decoding error. Quantization has been chosen because it enables to optimize the trade-off between average distortion (here, $d/2$) and the minimum alteration leading to a decoding error (here, $d/4$).

If the expected bit b and the decoded bit b' are the same, we say that the polygon matches. We maintain two counters, m (match) and t (total). The first one is incremented each time a polygon satisfying the watermarking conditions is found. The second one is incremented each time this polygon matches. Hence, the detection ratio m/t is the ratio of matching polygons.

It is easy to see that on a third party dataset, the probability that each polygon matches is $1/2$. Therefore, the ratio m/t is compared to its expected value $1/2$ to decide whether the mark of the owner is present in the document or not. Practically, a detection threshold α must be set to bound the detection area. We detect a mark when $|m/t - 1/2| \geq \alpha$. The relevance of the detection process highly relies on the value of α. Prop. 1 gives the detection threshold for a maximum false positive occurrence probability f_p.

Proposition 1. *(direct application of [10]) Let p the number of polygons satisfying the watermarking conditions. If each polygon has a probability* $1/2$ *to match, the probability* $\mathcal{P} = Pr\left(m/t - 1/2 \geq \alpha\right)$ *is such that* $\mathcal{P} \leq e^{-2\alpha t^2}$.

Then, the false positive occurrence probability defined as $f = \mathcal{P}\left(|m/t - 1/2| \geq \alpha\right)$ is such that $f \leq 2e^{-2\alpha t^2}$. Choosing $\alpha = -\log(\delta/2)/2t$ as the detection threshold,

permits to keep f under δ. We use this formula in our experiments to keep false positives occurrence probability under $\delta_0 = 10^{-4}$.

Algorithm 2: Detection algorithm

Input: secret key $\mathcal{K}$, watermarking ratio $1/\gamma$, h, quantization step interval $D = [d_{min}, d_{max}]$, max. false positive occurrence probability f_p
Data: (R', DB'), a suspect dataset
Output: MARK or NO_MARK
foreach *building* P *in* DB **do**
  $O \leftarrow$ `centroid`(P);
  $id \leftarrow$ `hsb`(O, h);
  `seed`$(G, \mathcal{K} \cdot id)$;
  if `nextInteger`$(G) \mod \gamma = 0$ **then**
    t++; `/* increment total count */`
    $\boldsymbol{u} \leftarrow$ `orientation`(P);
    $x_{max} \leftarrow \max\{p \in P | \mathbf{Op} \cdot \boldsymbol{u}\}$;
    $d \leftarrow d_{min} +$ `nextFloat`$(G).(d_{max} - d_{min})$;
    $b \leftarrow$ `nextInteger`$(G) \mod 2$; `/* expected bit` b `*/` ;
    $x'_0 \leftarrow$ `quantize`$(x_{max}, d, 0)$; `/* closest 0-quantizer */`
    $x'_1 \leftarrow$ `quantize`$(x_{max}, d, 1)$; `/* closest 1-quantizer */`
    if $|x_{max} - x'_0| > |x_{max} - x'_1|$ **then**
      $b' \leftarrow 1$; `/* found bit is` $b' = 1$ `*/`
    else
      $b' \leftarrow 0$; `/* found bit is` $b' = 0$ `*/`
    if $b = b'$ **then** m++; `/* increment match count */`
$\alpha \leftarrow$ `threshold`(f_p, t);
if $|m/t - 1/2| > \alpha$ **then return** MARK; **else return** NO_MARK;

4 Experiments

4.1 Framework

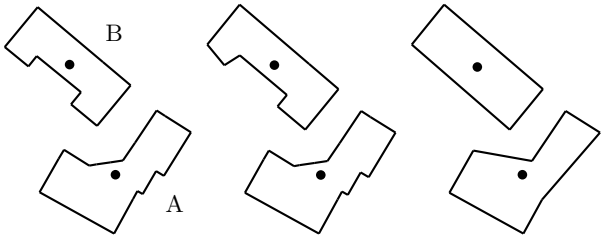
Data All experiments presented in this paper (except speed ones) were realized on buildings from the French city of Pamiers. The data is part of the **BD TOPO**®[12], a topological database product from the French National Mapping Agency (IGN), the major maps provider on the French market. The product consists of several coherent layers (hydrographic network, roads, buildings...) from which we extracted only the buildings layer. This layer is composed of 4 278 polygons (35 565 vertices), representing dense build areas (downtown – west side) as well as sparse ones (residential blocks – east side). It has a contractual accuracy of 1 meter. A glimpse on the Pamiers dataset is given in Fig. 6.

Fig. 6. Pamiers buildings (extract)

Software We developed a Java version of our algorithm and packaged it as a library for our generic database watermarking framework [17]. The dataset was stored in a Postgresql/Postgis database. Advanced geographic functions from GeOxygene [15] were also used. GeOxygene is an open source application developed at IGN. It aims at providing an open framework which implements OGC/ISO specifications for the development and deployment of geographic applications.

Filter/Attacks We performed an extensive series of experiments to validate the robustness of our method. All the filters/attacks presented in Section 2.3 were tested.

Protocol We consider that an attack is successful if it destroys the watermark with high probability while inducing a quality loss comparable to the one introduced by the watermarking process. In a same manner, we consider that a watermarking algorithm A is better than an algorithm B against a specific attack if it is as robust as B while inducing a quality loss significantly smaller than B. To emphasize the benefits of watermarking by expansion, we put side by side a random noise based method and ours and compare them in terms of robustness/distortion trade-offs. So, we begin by quantifying the quality losses of both schemes.



(a) Original (b) AKH (c) DP($d = 5m$)

Fig. 7. Examples of filters

4.2 Impact of Watermarking on Quality

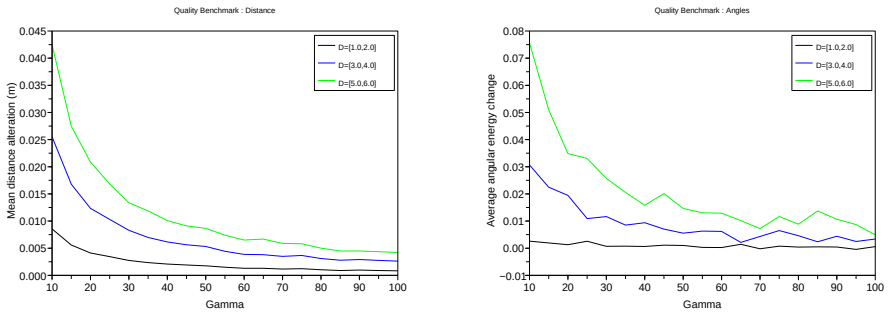
To evaluate the impact of our watermarking algorithm on the Pamiers dataset, we applied it with different watermarking ratios $1/\gamma$, with $\gamma \in \{10, 15, \ldots, 100\}$, and different quantization ranges $[d_{min}, d_{max}] \in \{[1, 2], [3, 4], [5, 6]\}$. These ranges start from data accuracy (1 meter) to the maximum reasonable alteration (6 meters).

Average Accuracy Alteration The impact on the mean accuracy is displayed on Fig. 8(a). Alteration increases when quantization steps increase and when γ decreases. We observe that the mean alteration is proportional to $(d_{min} + d_{max})/\gamma$. For instance, when $[d_{min}, d_{max}] = [3, 4]$, a good approximation of the mean accuracy alteration is $0.07.(d_{min} + d_{max})/\gamma$. The ratio $1/\gamma$ is not surprising since on the average, $1/\gamma$ polygons are watermarked. Furthermore, the expected alteration for the farthest vertex from the centroid is $(d_{min} + d_{max})/4$. The alteration of all the points from a watermarked polygon are proportional to the alteration of this particular vertex. These dependencies can be used by the watermarker to choose the parameters of the marking algorithm: if d_{max} is obtained as the maximum allowed alteration, and if the target mean alteration is fixed, one can easily compute γ.

Maximum Accuracy Alteration The behavior is similar with the one obtained for average accuracy alteration.

Angular Quality We verify that the variation of angular quality introduced by our method is negligible on Fig. 8(b). Even for the highest quantization steps, $[d_{min}, d_{max}] = [5, 6]$, the highest angular energy variation is at most $+0.08$. As a comparison, a weak gaussian noise (deviation $d = 0.2m$) increases the angular energy by $+6.19$.

Area Modification The impact of the watermarking algorithm on the areas of the polygons of the dataset is displayed on Fig. 9. On Fig. 9(a) is displayed the maximum relative surface alteration. Its is proportional to the mean of minimum



(a) Average accuracy alteration (b) Average angular energy alteration

Fig. 8. Impact of watermarking

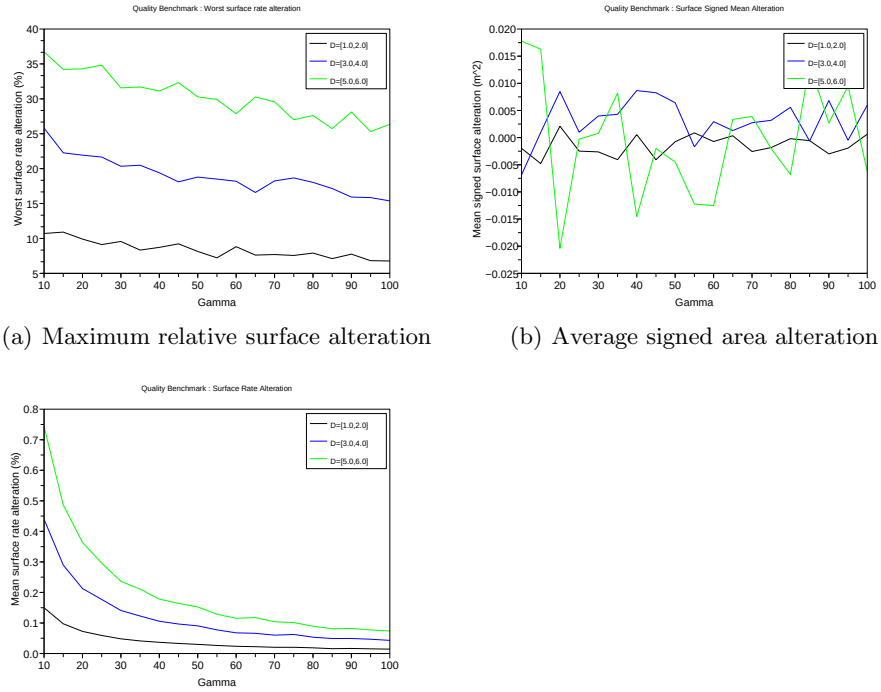
and maximum quantization steps. On Fig. 9(b) is displayed the average signed area alteration. The alterations of surface compensate themselves. On Fig. 9(c) is displayed the average relative area alteration. Its behavior is similar with the one observed for average accuracy alteration.

Detection: False Positives On Fig. 10 are displayed the detection ratios obtained with different secret keys on a previously watermarked dataset. A hundred different keys were tested, among which the one used for watermarking. Detection threshold $\alpha \approx 15\%$ was chosen so that false positive occurrence probability is at most 10^{-4}. It appears that only the secret key $\mathcal{K} = 100$ used in the embedding process leads to a positive detection of mark. This experimentally validates the relevance of the detection algorithm.

Watermarking Speed To evaluate the watermarking rate our implementation can achieve, we used a significantly larger dataset, still extracted from the **BD TOPO**®. It consists of the building layer from the Pyrénées Orientales (64), a French department. It contains 243 201 polygons (1 880 405 points and 65 megabytes for the geometry). Using $\gamma = 20$, the whole dataset was watermarked in 74 mins 30s (including database I/O times),i.e. $>$ 50 polygons per second. The detection process took 1min 38s. A notebook powered by an Intel Core Duo processor running at 1.6Ghz with 2Gb of RAM and a 7200rpm hard drive was used for this experiment.

4.3 Random-Noise Based Watermarking

The random noise scheme is based on the insertion of a pseudo-random noise within the data. Even if it is not a scheme found in the literature, it supersedes existing ones and mimic the behavior of others. It is similar, in most cases, to the schemes that introduce perturbation in the dataset without taking into account the shapes of the polygons [19]. The point is that if shapes are expected



(a) Maximum relative surface alteration

(b) Average signed area alteration

(c) Average relative area alteration

Fig. 9. Impact of watermarking on surface

to be regular, a watermarking algorithm working by altering these shapes is suboptimal on terms of robustness/distortion trade-off. Indeed, most users will correct the shapes so that the regularity aspect is brought back. This operation is very likely to remove the watermark.

The algorithm consists of looping over all the vertices of the dataset. The x and y coordinates of the vertices are watermarked independently. For each coordinate, a PRNG is seeded with its highest significant bits. Only the least significant bits are altered. Their positions and values are determined according to the drawings of the PRNG. This method can be seen as a straightforward extension of the existing AKH [3] scheme in which most significant bits of the coordinates are used as primary keys and least significant bits as alterable attributes.

The quality loss introduced by such a random noise scheme is controlled by three parameters: the watermarking ratio $1/\gamma$, the least highest significant bit $lspow2$ and the number $\xi = 2$ of alterable powers of two. For instance, if $lspow2 = 2$ and $\xi = 2$, the maximum distortion on a coordinate is $2^{lspow2-1} = 2$ and the minimum distortion is $2^{lspow2-\xi} = 1$. Using higher values of ξ enables for extra embedding bandwidth but lead to distortions that are removed by

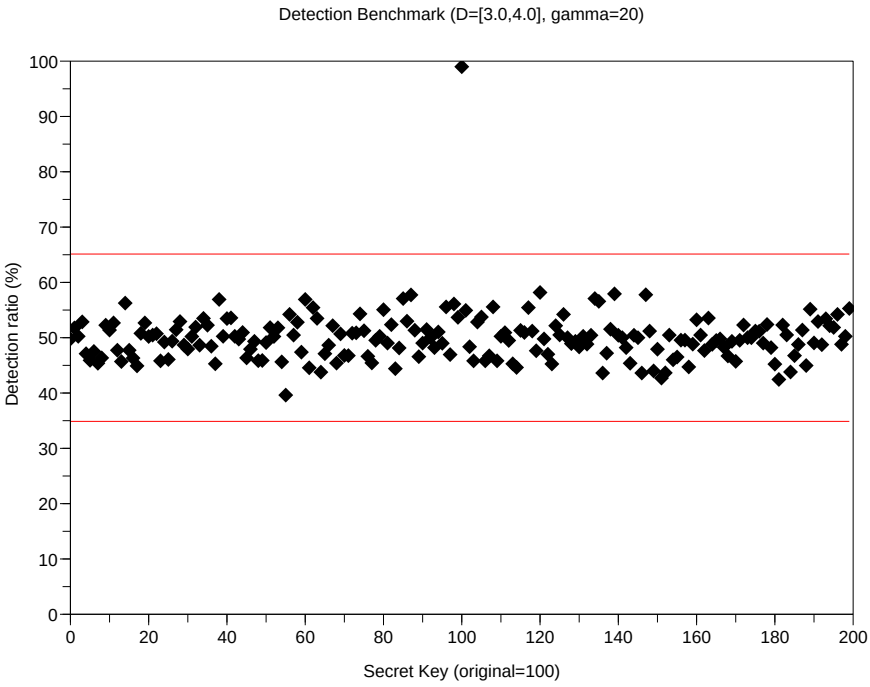
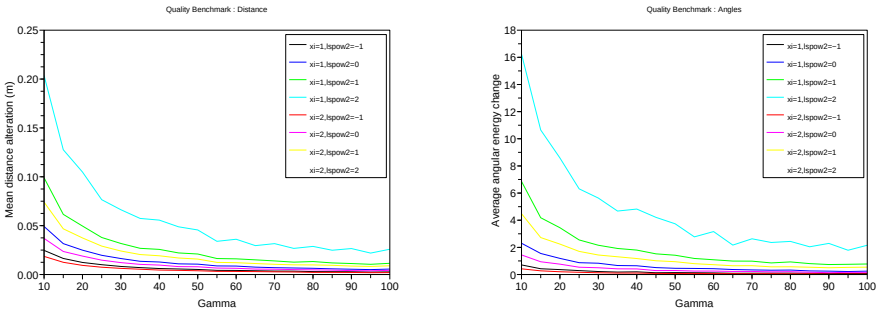


Fig. 10. Undetectability of the mark without the correct key

any rounding of the coordinates. In what follows, we compare our scheme (with $d_{min} = 3$ and $d_{max} = 4$) with two random noise schemes parameterized with two sets of parameters. For the first one, RNW1, $lspow2 = -1, \xi = 1$; for the second one, RNW2, $lspow2 = 2, \xi = 2$. These values were chosen for the following reasons: RNW1 achieves an average accuracy alteration (compare Fig. 8(a) and Fig. **??** - both curves decrease as $1/\gamma$ and have a mean accuracy alteration of 0.025 for $\gamma = 10$) very close to the one observed using our method whereas RWN2 has a maximum alteration on each vertex equals to ours.



(a) Average accuracy alteration (b) Average angular energy alteration

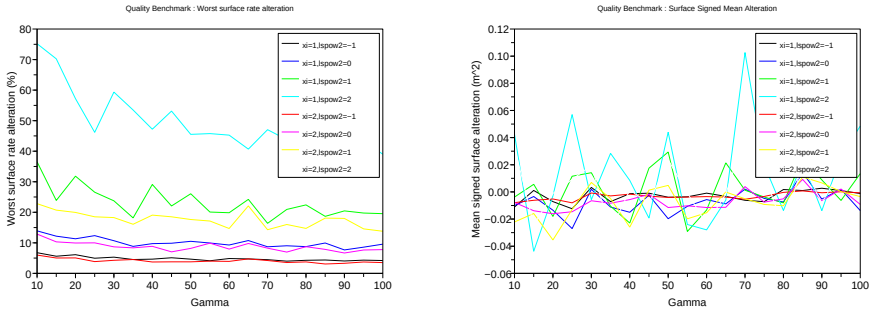
Fig. 11. Impact of random noise watermarking

4.4 Robustness Filters/Attacks

For all the experiments, we present the detection ratios observed after application of a filter, together with the detection threshold of the mark (computed for a false positive occurrence probability of 10^{-4}). We put side by the results obtained using our method, RNW1, RNW2 and the combined scheme presented in Section 4.5.

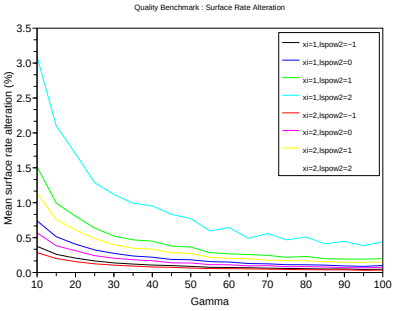
Squaring We present on Fig. 13, the detection ratios observed after squaring. Three different values of the maximum allowed point position alteration were tested, namely $d = 1.0$ (commonly used value), $d = 1.5$ and $d = 2.0$ (very aggressive squaring, used here only for validation). For our scheme, squaring has no effect on the detection ratio no matter the watermarking parameters. Even in a worst case scenario, $\gamma = 100$ and $d_{max} = 2$, the watermark is still detected. For RNW1 and RWN2, the detection ratio is considerably lowered when the data is squared.

Douglas & Peucker Simplification The Douglas-Peucker algorithm [6] is a polyline simplification algorithm. It is systematically used by geographic data users



(a) Maximum relative surface alteration

(b) Average signed area alteration



(c) Average relative area alteration

Fig. 12. Impact of watermarking on surface

with a small factor to filter points of the database. It consists of pruning the points of a polyline whose distance to the line joining the polyline bounds is too small. The distance threshold under which these points are pruned is called d. The higher this threshold, the more aggressive the algorithm is. We tested the robustness of our algorithm against Douglas-Peucker filtering for different threshold values. An example of Douglas-Peucker simplification with $d = 5$ meters is given on Fig. 7(c). The results are displayed on Fig. 14. Note that our algorithm performs very well when the Douglas-Peucker filtering distance is 1 or 2 meters. The detection ratio never falls below the detection threshold. The reason is that the shapes of the polygons are regular enough so that they are invariant to these filters. For a filtering distance of 5 meters, the mark is removed in some situations. But such a filtering is very aggressive: it can remove a $4m \times 4m$ room in a building.

Cropping Observe that the whole dataset is not necessarily interesting for a malicious user, since profit can still be made from the (illegal) redistribution of a subset. Our method does not require the whole suspect dataset to perform detection. This is illustrated on Fig. 15 for our scheme. We randomly generated rectangular subsets of the watermarked dataset and performed detection

on them. On the x-axis are displayed the number of polygons in a crop and on the y-axis the number of watermarked and matching polygons. We also added the minimum number of matching polygons that must be found to detect the watermark, i.e. the detection threshold. In all cases the watermark is detected. Furthermore, the curves are, once removed local artifacts, almost linear. This indicates that the distribution of the watermarked polygons over the dataset is nearly uniform. For other schemes, experiments not presented here show that the repartition of watermarked polygons is nearly uniform so detection is not too much affected by squaring.

Gaussian Noise Gaussian noises with deviations of $20cm$, $60cm$ and $1m$ were tested and the results displayed in Fig. 16. For reasonable noises ($d = 20cm$ or $d = 60cm$), marks are still detected. When $d = 1m$, the watermark is removed for a large interval of γ.

Interestingly, the application of a squaring algorithm on a noised dataset increases the detection ratio for our scheme. It even permits the recovery of the watermark for higher values of γ. RNW1, RNW2 and the combined scheme show similar behaviors.

Applying a New Watermark On Fig. 17(a), we show how the detection algorithm copes with a new application of the watermarking algorithm with a different key. Different watermarking parameters for the attack were tested: highest quantization steps and lowest γ. For all parameters, the observed detection threshold is such that the first watermark is still detected. Applying another watermark has a limited effect on the first watermark. This is due to the fact that the two watermarks are embedded into distinct polygons. Moreover, on the sparse polygons chosen simultaneously by the two watermark embeddings, the second watermark is the only one to be detected on these polygons. RNW1, RNW2 and the combined scheme show similar behaviors.

Enlarge to rectangle filter For this experiment, we used the Map Generalization Toolbox of OpenJump [2]. The enlarge to rectangle filter consists of replacing each building of the max with its minimum bounding rectangle. If a map is to be drawn from the geodata, its scale can be passed as an argument to the enlarge to rectangle filter. In this case, illegible buildings are replaced by their minimum bounding rectangle and enlarged so that they are visible. Larger buildings are not modified. The threshold values are chosen according to the Swiss Society of Cartography [20], e.g. 0.25mm for width and 0.35mmx0.35mm for buildings minimum size on maps. The results of the experiments are presented on Fig. 18. They show that replacing the all buildings by their bounding rectangle washes the watermark. When a watermarked dataset is used for map generation, the detection ratio decreases as the scale increases. This is not surprising since a high scale implies that many buildings are represented by a minimal area rectangle. So bits embedded in such buildings can not be recovered. Nevertheless, it is interesting to notice that watermarking one polygon out of ten (i.e. $\gamma = 10$)

enables to recover marks in a map with scale 25000. Compare to RNW1 and RNW2, our scheme performs better.

Elongation Change Filter The elongation change filter consists of multiplying the main length of all polygons with a fixed factor. For experiments presented on Fig. 19, ratios 0.85,0.9,...,1.2 were used. Hopefully, when the ratio is 1, the detection ratio is close to 1. If the ratio is between 0.95 and 1.05, the watermark is still detected. This means that an attacker who wants to ensure that the watermark is removed must alter the main length of *all* polygons by more than 5%. Compared to RNW2 and RNW2, the performances of our scheme are not as good since elongation ratios 0.9 and 1.1 remove the watermark embedded using our scheme whereas the ones embedded using random noise based scheme are still detectable. This is not damning since the elongation change filter is the worse attack our scheme can encounter since it precisely affects the embedding area used for watermarking.

Minimum Bounding Rectangle Filter This filter replaces each polygon by its minimum bounding rectangle. This filter is very aggressive, too for both algorithms since the watermark is removed for a large range of γ in all cases. But, as it can be seen on Fig. 20, our scheme performs slightly better.

Missing Reference System Without a reference system, a dataset is useless for automatic operation and interoperability. Meanwhile, the suspect dataset can be mapped into the original reference system to perform the detection process. This mapping operation is quite common and easy, as soon as (a reasonably large part of) the original dataset is available.

Other Attacks Many other kinds of attacks may be envisioned. The mixing attack consists of mixing a portion of a watermarked dataset with an unwatermarked one. This attack is expected to lower the detection ratio but not render the watermark unreadable.

The aliasing attack consists of switching the edges of the polygon with zig-zag shaped sequences of edges. The goal is to modify the orientation of the polygon. But it implies adding a huge number of extra (fake) points to the database and altering the overall shapes of polygons. Furthermore, the artifact needs to have an amplitude exceeding angle tolerance ratio.

Note also that our method is invariant to polygons rotation since the expansion coefficient is defined relatively to polygon orientation and not to a particular reference system.

We also observe that, for our scheme, squaring the data prior to the watermarking process, beside increasing its value, enhances slightly the detection ratio.

4.5 Discussion

Table 2 sums up robustness experiments. Robustness results for our method are presented in the WM column. Whether a method achieves robustness is

indicated by with the appropriate Yes/No answer. When a method is robust as far as enough polygons are watermarked, the threshold watermarking ratio under which the method has been experimentally proved robust is also indicated. Remark that robustness might also be achieved for higher values of γ. In the second and third columns are shown the average quality loss introduced by the attacks. These must be put into balance with the distortion introduced by watermarking algorithms (see Section 4.2 and 4.3).

As we expected, our method resist all kinds of squaring, including the most destructive ones. On the contrary, watermarks embedded using random-noise based algorithms are washed out. Our method shows also robustness against the majority of attacks. The fact that GN(d=1m) and DP(d=5m) erase the watermark must be put into balance with the quality losses these attacks introduce. They tremendously reduce the quality of the dataset. Furthermore, it performs relatively against change elongation filters since the length of polygons must be increased or decreased by more than 10% to ensure watermark removal. But this remains the weakest point of our method since the embedding area is directly affected.

Compared to random-based schemes, our algorithm performs better in the majority of cases. Whereas RWN1 and RWN2 are more robust against the change elongation filter, they have a significantly higher impact on quality compared to our method. In that sense, we are confident to say that our algorithm for building watermarking achieves a very good distortion/robustness trade-off.

In the last column of the table, we combined our scheme and RNW2. In a first time, we apply our scheme and then apply RNW2 on the centroid of the polygon (left invariant by the first step). We obtain a modification of the coordinates of the centroid which is used to translate the polygon. Obviously, this combined scheme introduces a larger quality loss but it achieves robustness for all the filters/attacks used in the paper. We think that this combined scheme is a relevant practical scheme.

5 Related Work

In our database approach, polygons are stored in a relational database management systems enriched with geographical features. Since polygons are stored in relational tables, state-of-the art watermarking algorithms for relational databases might have been used. It happens that least significant bits modifications used in previous works [3] can not be mapped onto our geographical setting. It is easy to alter least significant bits of points of the map but the angular quality is not taken into account (on the contrary, least significant bits methods may perform well on simple points databases, like point-of-interest datasets in use in GPS viewers). Methods described in [9, 26] allow for the description of usability queries to be preserved by watermarking. But they either focus on basic numerical aggregates like `SUM` queries [9], which are not rich enough to represent angular constraints, or based on a trial and error method to handle generic black-box

Table 2. Robustness of watermarking

Filter	Precision alt.	Angular energy	WM	RNW1	RNW2	Combined
SQ (d=1m)	0.19	-14.1	Yes	No	Yes	Yes
SQ (d=1.5m)	0.23	-16.2	Yes	No	$\gamma < 90$	Yes
SQ (d=2m)	0.27	-17.2	Yes	No	$\gamma < 80$	Yes
DP (d=1m)	N/A	- 1.3	Yes	Yes	Yes	Yes
DP (d=2m)	N/A	- 0.3	Yes	Yes	Yes	Yes
DP (d=5m)	N/A	68.1	$\gamma < 70$	Yes	Yes	$\gamma < 100$
CA	0	0	Yes	Yes	Yes	Yes
GN (d=0.2m)	0.18	6.19	Yes	$\gamma < 30$	Yes	Yes
GN (d=0.6m)	0.53	40.00	Yes	No	$\gamma < 75$	Yes
GN (d=1m)	0.89	78.52	$\gamma < 65$	No	$\gamma < 30$	$\gamma < 75$
OW	-	-	Yes	Yes	Yes	Yes
ETR	N/A	-6.53	$\gamma < 40$	No	$\gamma < 25$	$\gamma < 50$
ETR(scale=25000)	N/A	-6.06	No	No	No	$\gamma < 40$
ETR(scale=250000)	N/A	-0.03	Yes	Yes	Yes	Yes
CE(scale=0.90)	1.06	0.16	No	No	$\gamma < 45$	Yes
CE(scale=0.95)	0.53	0.02	$\gamma < 95$	No	Yes	Yes
CE(scale=1.0)	0	0	Yes	Yes	Yes	Yes
CE(scale=1.05)	0.53	0.09	$\gamma < 95$	Yes	Yes	Yes
CE(scale=1.10)	1.06	0.27	No	No	$\gamma < 50$	Yes
MBR	N/A	-6.6	$\gamma < 75$	$\gamma < 35$	$\gamma < 50$	$\gamma < 75$

queries [26]. Using the latter method, it might not be possible to reach a valid set of alteration since no search strategy is defined.

Despite that state-of-the art is very rich on watermarking still images which can be directly applied to image maps, fewer works were carried on on watermarking vector maps. A complete study of vector maps watermarking [19] has been recently published which divides this field into three categories: spatial domain watermarking, transform domain (DCT,DWT,..) watermarking and 3D meshes adapted watermarking algorithm. Spatial domain watermarking consists of modifying directly the coordinates of the vertices of the dataset. Patch-based techniques [13, 21, 24, 29] are based on a decomposition of the dataset into patches in which bits are embedded by vertices translation [21] and/or data distribution within a patch [13, 24, 29]. Such schemes are sensitive to patch decomposition that can vary when the dataset is cropped. Schemes of [13, 24] create detectable nodes aggregations; [21] is not blind and [29] does not respect the shape of smooth objects (including squared buildings).

Vertices addition based [11, 18, 23] techniques are the best from a quality viewpoint. They consists of interpolating the existing edges of the dataset with fake points. In the context of buildings where the shapes are regular, such interpolations are easily detected and removed by any simplification algorithm. In our context, these schemes are not robust enough. The least-significant bit watermarking method presented in [28] preserves accuracy but is very sensitive to vertices addition, which destroys synchronization.

Transform domain techniques apply on polylines preferably to closed polygons. They are complementary approaches of ours. For instance, [8] and our method can be simultaneously used in a multiple layer database to improve robustness.

In [25], a high watermarking capacity algorithm for vector maps is introduced. It is based on a previous work on 3D meshes watermarking by the same authors. It is robust against common geographical filters like Douglas-Peucker simplification algorithm. It is based on a decomposition of the database into patches and by moving points of a common patch into a subpatch to embed the watermark. Nevertheless, efficient attacks erasing the watermark without destroying the quality of the document can be easily planned, as the method requires known synchronization points.

6 Conclusion

In this paper we presented a blind watermarking algorithm for polygonal datasets. It is well suited to building layers of geographical datasets since watermarks are invariant through aggressive geographical filters applied by data users. We experimentally showed that it is difficult for an attacker to erase the watermark without paying an extra quality fee, compared to watermarking. Indeed, the possibility that an attacker destroys the watermark must be put into balance with the quality loss it introduces. The algorithm has been implemented into an open database watermarking framework [17]. We are currently working on designing algorithms for other layers of geographical datasets. A real challenge we are faced with is to deal with the interactions between the different layers. Indeed, watermarking algorithms must be adapted to the data; there is no unique solution. Even if we know how to perform watermarking and detection on a single layer, it is challenging to orchestrate several algorithms on several layers so that resulting watermarked datasets remain consistent.

7 Acknowledgements

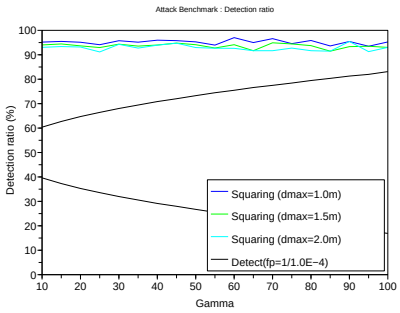
We deeply thank Eric Grosso from IGN for providing us with valuable advices on how to use the GeOxygene framework.

References

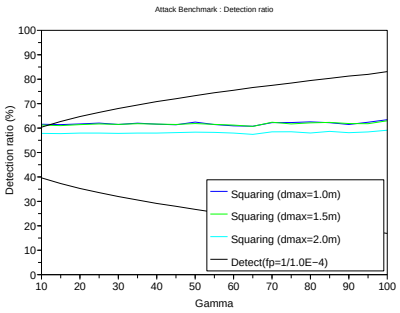
1. GéoPortail (visited 03/20/2007). `http://www.geoportail.fr`.
2. Openjump – the free, java based and open source geographic information system for the world. `http://www.openjump.org`.
3. R. Agrawal, P. J. Haas, and J. Kiernan. Watermarking relational data: framework, algorithms and analysis. *VLDB J.*, 12(2):157–169, 2003.
4. S. Airault. De la base de données à la carte : une approche globale pour l'équarrissage de bâtiments (in french). *Revue Internationale de Géomatique*, 6(2-3):203–217, 1996.

5. P. Bourke. Calculating the area and centroid of a polygon, July 1988. http://local.wasp.uwa.edu.au/~pbourke/geometry/polyarea/.
6. D. Douglas and T. Peucker. Algorithms for the reduction of the number of points required for represent a digitized line or its caricature. *Canadian Cartographer*, 10(2):112–122, 1973.
7. C. Duchêne, S. Bard, X. Barillot, A. Ruas, J. Trevisan, and F. Holzapfel. Quantitative and qualitative description of building orientation. In *Fifth workshop on progress in automated map generalisation, ICA, commission on map generalisation*, Apr. 2003.
8. H. Gou and M. Wu. Data hiding in curves with application to fingerprinting maps. *IEEE Transactions on Signal Processing*, 53(10):3988–4005, Oct. 2005.
9. D. Gross-Amblard. Query-preserving watermarking of relational databases and XML documents. In *PODS '03: Proceedings of the Twenty-Second Symposium on Principles of Database Systems*, pages 191–201. ACM, 2003.
10. W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.
11. W. A. Huber. Gis and stegranography - part 3: Vector stegranography, Apr. 2002. http://www.directionsmag.com/article.php?article_id=195.
12. Institut Géographique National. BD TOPO - descriptif technique (in french), december 2002. http://www.ign.fr/telechargement/MPro/produit/BD_TOPO/JT_Agglo/DT_BDTOPOPays_1_2.pdf.
13. H. I. Kang, K. I. Kim, and J. U. Cho. A vector watermarking using the generalized square mask. In *Information Technology: Coding and Computing,*, pages 234–236, Apr. 2001.
14. S. Katzenbeisser and F. A. Petitcolas. *Information hiding, techniques for steganography and digital watermarking*. Artech house, 2000.
15. C. laboratory. Geoxygene: an open framework for the development and deployment of gis applications.
16. J. Lafaye, J. Béguec, D. Gross-Amblard, and A. Ruas. Blind watermarking of geographical databases by polygon expansion. Technical report, Cnam-CEDRIC, Feb. 2007.
17. J. Lafaye, D. Gross-Amblard, M. Guerrouani, and C. Constantin. Watermill: an optimized fingerprinting system for databases under constraints. *submitted to TKDE*, 2007.
18. C. M. Lopez Vazquez. Method of inserting hidden data into digital archives comprising polygons and detection methods, November 2003. US Patent no. 20030208679.
19. X. Niu, C. Shao, and X. Wang. A survey of digital vector map watermarking. *International Journal of Innovative Computing, Information and Control*, 2(6), Dec. 2006.
20. S. S. of Cryptography. vol. 17: Topographic maps – map graphic and generalization.
21. R. Ohbuchi, R.Ueda, and S. Endoh. Robust watermarking of vector digital maps. In *Multimedia and Expo, 2002. ICME '02*, volume 1, pages 577– 580, 2002.
22. R. Ohbuchi, H. Ueda, and S. Endoh. Watermarking 2D vector maps in the mesh-spectral domain. In *Shape Modeling International*, pages 216–228. IEEE Computer Society, 2003.
23. K. T. Park, K. I. Kim, H. I. Kang, and S.-S. Han. Digital geographical map watermarking using polyline interpolation. In *PCM '02: Proceedings of the Third IEEE Pacific Rim Conference on Multimedia*, pages 58–65, London, UK, 2002. Springer-Verlag.

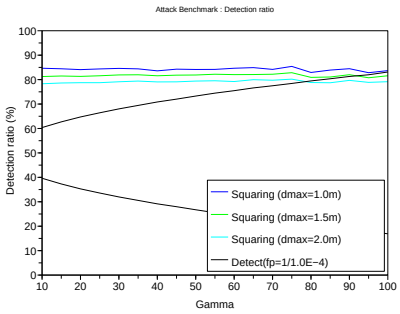
24. M. Sakamoto, Y. Matsuura, and Y. Takashima. A scheme of digital watermarking for geographical map data. In *Symposium on cryptoraphy and information security*, Okinawa, Japan, Jan. 2000.
25. G. Schulz and M. Voigt. A high capacity watermarking system for digital maps. In *MM&Sec '04: Proceedings of the 2004 workshop on Multimedia and security*, pages 180–186, New York, NY, USA, 2004. ACM Press.
26. R. Sion, M. J. Atallah, and S. Prabhakar. Rights protection for relational data. *IEEE Trans. Knowl. Data Eng.*, 16(12):1509–1525, 2004.
27. The International Hydrographic Organization (IHO). *Specifications for Chart Content and Display Aspects of ECDIS*. IHO, 5th edition, december 2001.
28. M. Voigt and C. Busch. Watermarking 2d-vector data for geographical information systems. In *SPIE, Security and Watermarking of Multimedia Content*, volume 4675, pages 621–628, 2002.
29. M. Voigt and C. Busch. Feature-based watermarking of 2d vector data. In *SPIE, Security and Watermarking of Multimedia Content*, volume 5020, pages 359–366, June 2003.



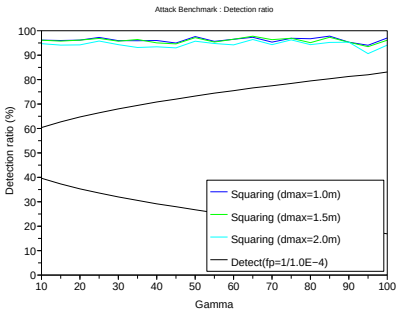
(a) WM



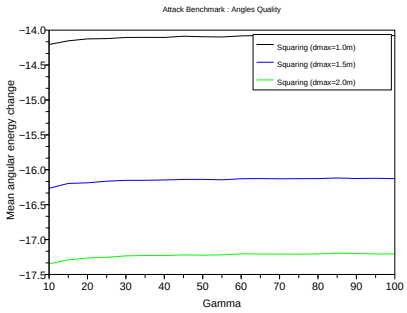
(b) RNW1



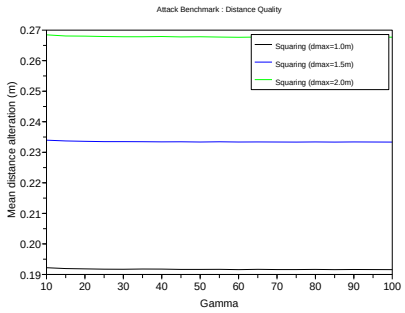
(c) RWN2



(d) Combined

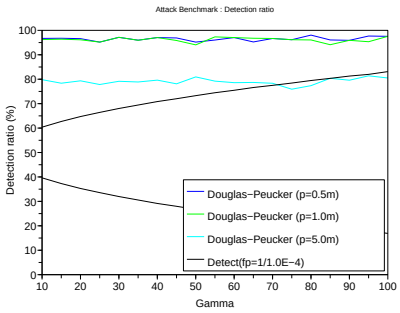


(e) Angles quality

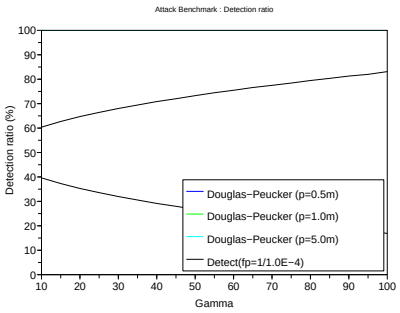


(f) Accuracy alteration

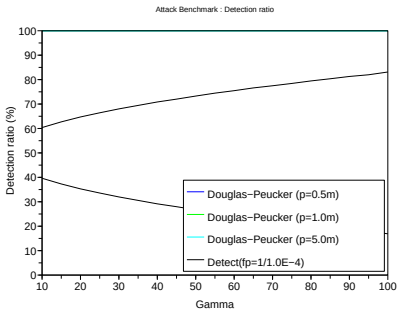
Fig. 13. Squaring filter



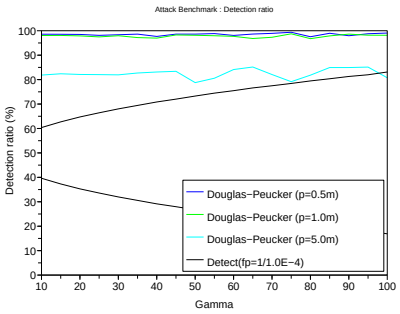
(a) WM



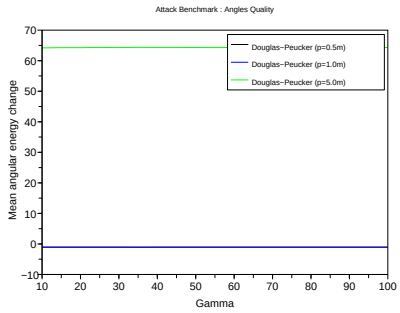
(b) RNW1



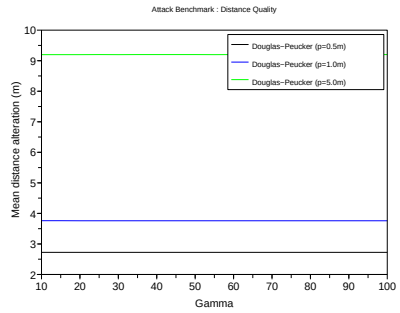
(c) RWN2



(d) Combined



(e) Angles quality



(f) Accuracy alteration

Fig. 14. Douglas-Peucker simplification

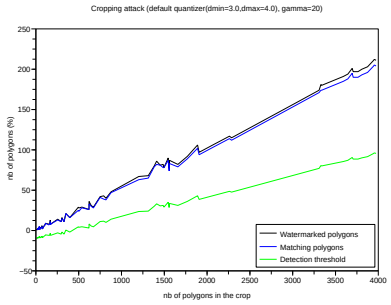
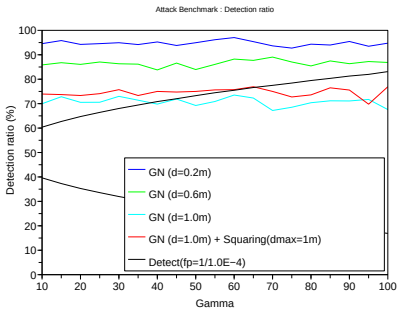
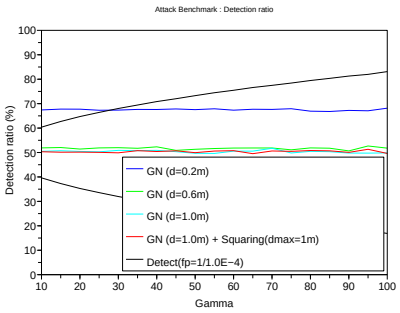


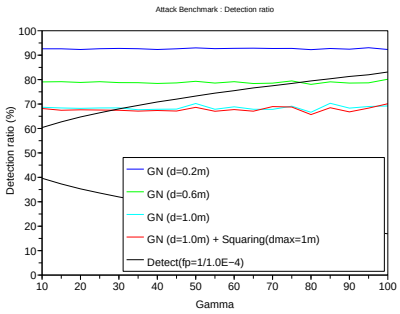
Fig. 15. Cropping attack



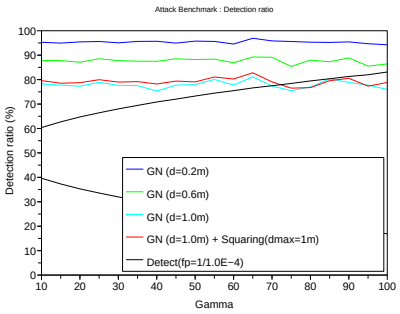
(a) WM



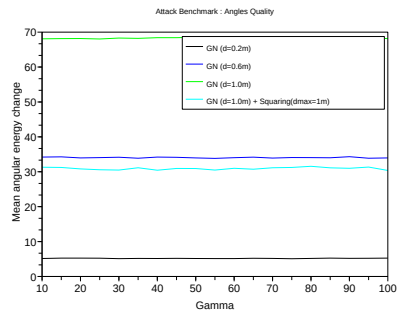
(b) RNW1



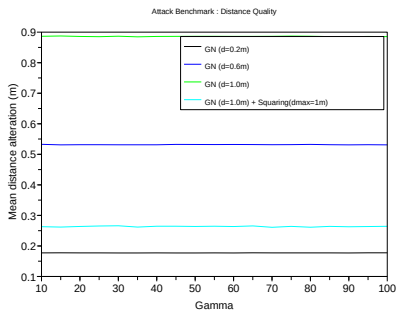
(c) RWN2



(d) Combined

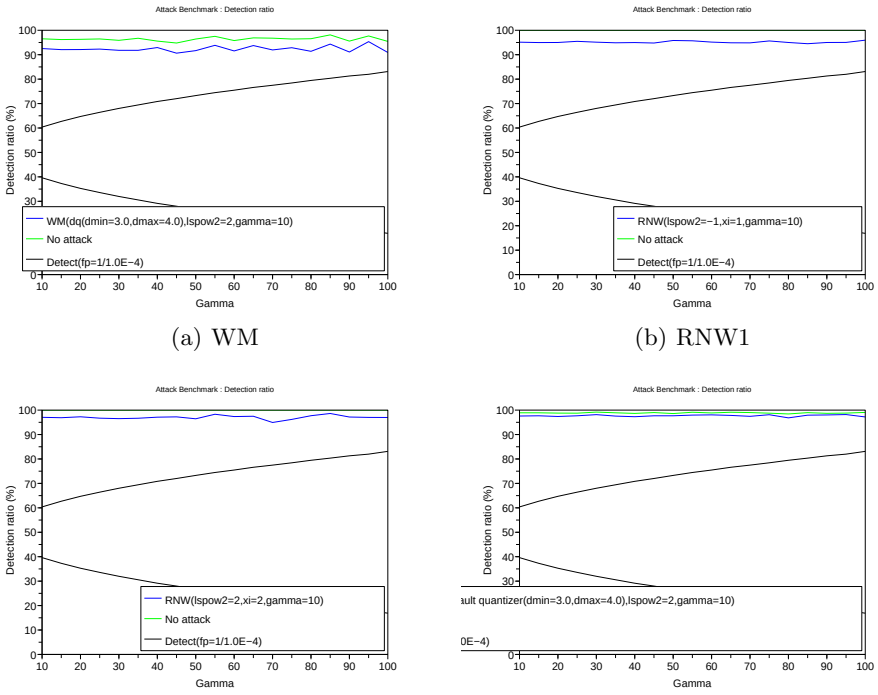


(e) Angles quality



(f) Accuracy alteration

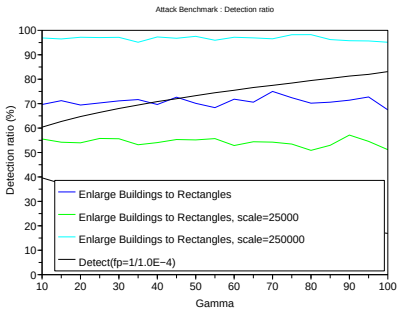
Fig. 16. Gaussian noise



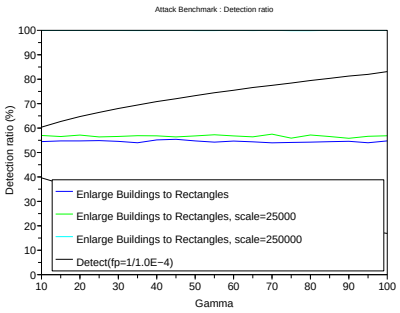
(a) WM (b) RNW1

(c) RWN2 (d) Combined

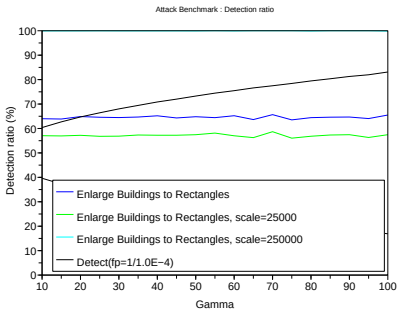
Fig. 17. Over-watermarking



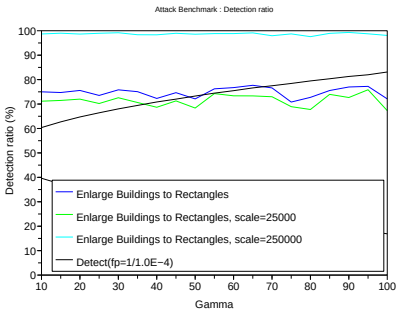
(a) WM



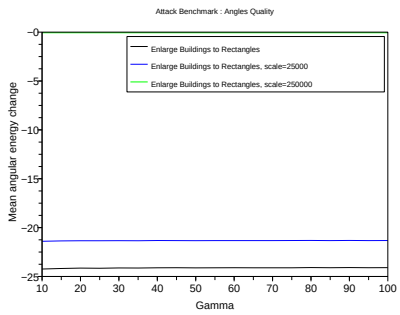
(b) RNW1



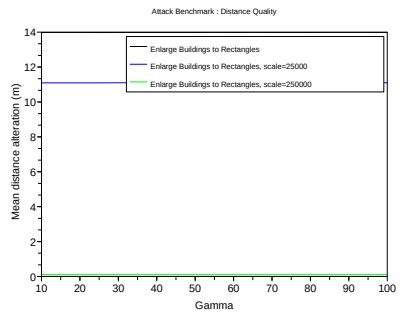
(c) RWN2



(d) Combined

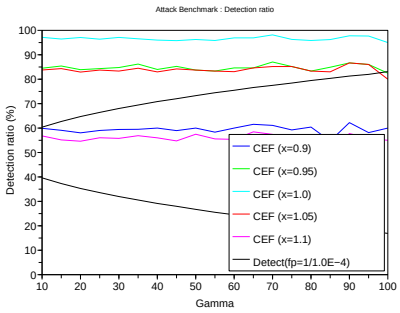


(e) Angles quality

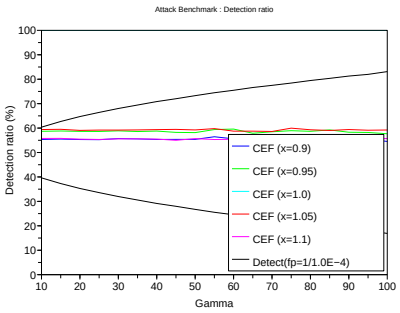


(f) Accuracy alteration

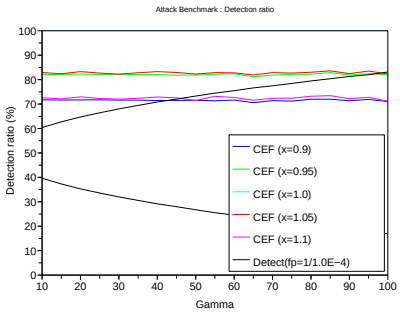
Fig. 18. Enlarge to rectangle filters



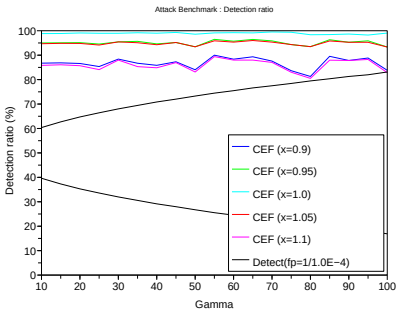
(a) WM



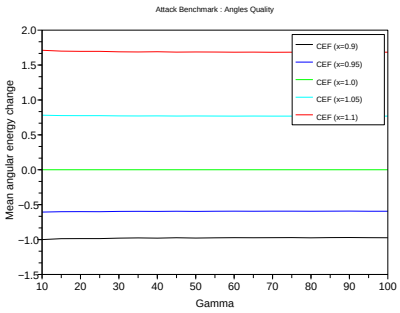
(b) RNW1



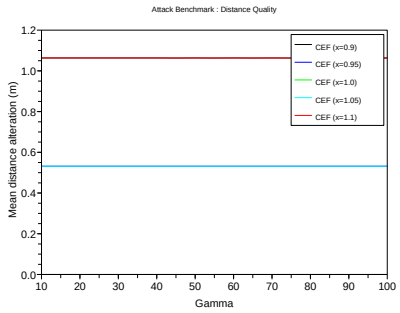
(c) RWN2



(d) Combined

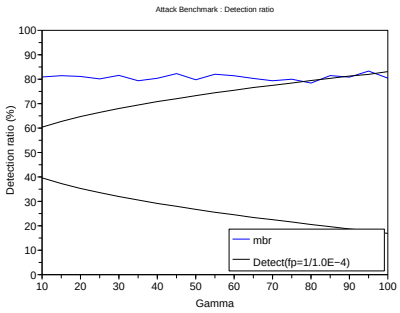


(e) Angles quality

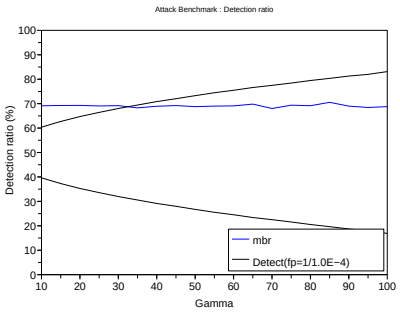


(f) Accuracy alteration

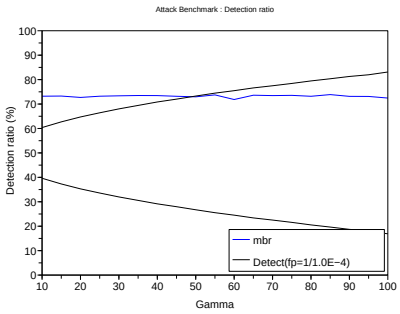
Fig. 19. Change elongation filter



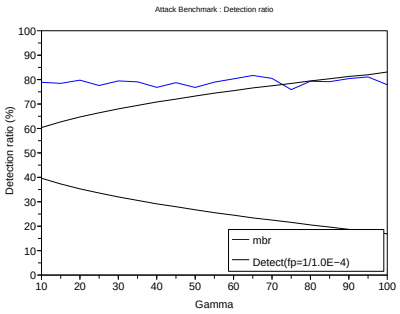
(a) WM



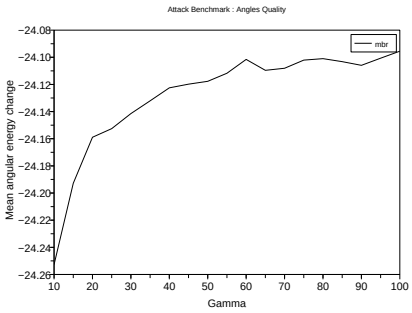
(b) RNW1



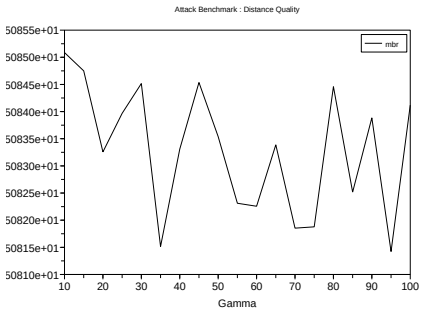
(c) RWN2



(d) Combined



(e) Angles quality



(f) Accuracy alteration

Fig. 20. Minimum bounding rectangle filter