International Journal of Multimedia and Ubiquitous Engineering
Vol.11, No.2 (2016), pp.265-276
http://dx.doi.org/10.14257/ijmue.2016.11.2.26

# RST Invariant Reversible Watermarking for 2D Vector Map

Nana Wang, Xiangjun Zhao and Chunli Xie

*School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, P. R. China*
*wangnana_5@yahoo.com*

***Abstract***

*This paper presents a RST (rotation, scaling and translation) invariant reversible watermarking method for 2D vector maps. Firstly, the proposed algorithm selects two reference vertices to calculate the normalized quantization step. Then, for each vertex, the Euclidean distance between a reference vertex and the vertex is divided into equal segments using the normalized quantization step. According to the segment which the vertex is divided into, a watermark is embedded by moving the vertex within its corresponding segment in a revertible manner. This algorithm not only recovers the original content after watermark extraction, but also correctly extracts the embedded watermarks after RST transformations. In addition, to control the distortions introduced by watermark embedding, the embedding parameter is carefully selected. Theoretical analysis and experimental results show that the proposed scheme provides RST invariance property, and good reversibility, invisibility, computational complexity and data capacity.*

***Keywords**: reversible watermarking, RST invariance property, 2D vector map, quantization, distortion control*

## 1. Introduction

With the rapid development of public networks, transferring 2D vector maps via internet becomes more and more popular in recent years. However, using powerful available tools and equipment, it is very easy even for an amateur to illegally modify and copy these valuable data. It is desirable to develop a strong method to protect the copyright and the integrity of the 2D vector map content.

In the past few years, many new techniques and concepts based on watermarking have been introduced to protect the copyright [1-12] and verify the integrity [13-14] of the vector map content. In most cases of watermarking, the original content is distorted in an irreversible way. However, due to the required high-precision nature of vector maps, modifications to vector maps are generally undesired. To satisfy this requirement, reversible (also referred to as invertible, lossless, or distortion-free) watermarking techniques [4-12], which allow the decoder to recover the original content upon extraction of the embedded data, have been proposed.

In 2D vector map reversible watermarking, Voigt *et al*. [7] embed data by modifying the frequency coefficient in the integer discrete cosine transform (DCT) domain. Due to the realization in the frequency domain, controlling the embedding distortion in the spatial domain seems complex. In [8], two reversible data hiding schemes based on the idea of difference expansion [6] were proposed: one hides data by modifying the differences between adjacent coordinates, and the second by manipulating the manhattan distances between neighboring vertices. The two approaches have good invisibility in the maps with dense vertices whereas the performance could be seriously decreased for the maps whose coordinates exhibit low correlation. Zhou *et al*. [9] offered an algorithm that embeds secret bits by modifying the difference histogram which is established by the differences of neighboring vertices. In [10], the nonlinear scrambling approach is used in a reversible watermarking scheme for 2D vector maps. Although it can avoid the vector map from being illegally used by unauthorized users, the embedding distortion may be greater than the vector map's precision tolerance. Cao *et al*. [11] proposed a method that embeds data by recursively modifying the mean coordinate value of each coordinate group. Despite the high capacity and the robustness against simplified attacks, the reversibility may be decreased as the recursive iterations are increased. Another scheme was proposed by Wang *et al*. [12], where the virtual coordinates are exploited for obtaining high capacity, low computational complexity and good invisibility. However, it is not robust against RST attacks, which is also a drawback of the afore-mentioned schemes for 2D vector maps [7-12]. RST attacks may introduce great distortions to the embedded watermarks without greatly decreasing the usability of the watermarked vector maps in some applications. The lack of RST invariance property may limit these schemes' application scope.

To resist against RST attacks, and simultaneously embed large amount of data without introducing significant visual distortions, we propose a reversible watermarking method for 2D vector maps based on Wang and Wang's scheme [5]. Rather than embedding data into the coordinates of each sorted axis in the PCA (principal component analysis) -coordinate system, we select two reference vertices to calculate the normalized quantization step, divide the Euclidean distance between each vertex and a reference vertex into equal segments, and embed data into each vertex by moving it within its corresponding segment in a reversible manner. For controlling the distortions introduced by watermark embedding, the embedding parameter is carefully selected. The advantages of the proposed method include the following: (1) RST attacks can be resisted; (2) the invisibility, reversibility, computational complexity and data capacity are good.

The remaining sections are organized as follows. Section 2 briefly reviews the reversible watermarking method by Wang and Wang [5]. Section 3 explains our RST invariant reversible watermarking algorithm in detail. We present our experimental results and an analysis of the algorithm in Section 4. Conclusions are summarized in Section 5.

## 2. Wang and Wang's Reversible Watermarking Scheme

The basic idea of Wang and Wang's algorithm [5] is as follows. Given a list of vertices, the coordinates of each axis are first sorted to produce sorted coordinate lists; then every sorted coordinate list is divided into intervals, each of which contains three adjacent coordinates. Finally, $c$ ($c \geq 1$) secret bits are embedded into an interval by modifying the interval's state value.

The state value of an interval is defined as follows. Suppose $x_1$, $x_2$ and $x_3$ ($x_1 \leq x_2 < x_3$) are three adjacent $x$ coordinates in the sorted $X$-axis. The interval between $x_1$ and $x_3$ can be divided into $P$ ($P \geq 2$) equal subintervals. The index of the subinterval which the coordinate $x_2$ is located on indicates the interval's state value.

Denote the interval which contains the three adjacent $x$ coordinates ($x_1$, $x_2$ and $x_3$) in the sorted $X$-axis as $Q(x_1, x_2, x_3)$, and the data to be embedded as $w$ ($w \in \{0,1,\ldots, 2^c - 1\}$).

The embedding process is described as follows.

Step 1. Calculate the r state of the interval Q,

$$\begin{cases} r = 0 & if \;\; x_2 < (x_1 + x_3)/2 \\ r = 1 & if \;\; x_2 \geq (x_1 + x_3)/2 \end{cases}.$$

(1)

Step 2. Divide the interval $Q$ into $2^{c+1}$ equal subintervals, and calculate the length of each subinterval $l_s$ by

$$l_s = (x_3 - x_1) \big/ 2^{c+1} \quad .$$

(2)

Step 3. Compute the state value s (s $\in \{0,1,\ldots, 2^{c+1} - 1\}$) of the embedded interval Q'($x_1$', $x_2$', $x_3$'), *i.e.*, the index of the subinterval which the embedded coordinate $x_2$' should be located on,

$$s = 2^c \times r + w.$$

(3)

Step 4. Obtain the embedded $x_2$' by moving $x_2$ to the $s$-th subinterval,

$$x_2' = x_1 + s \times l_s + \kappa / 2^c,$$

(4)

Where

$$\kappa = x_2 - (1 - r) \times x_1 - r \times (x_1 + x_3)/2.$$

(5)

Since the data w is embedded by moving $x_2$ between $x_1$ and $x_3$, $x_2$' remains within the range [$x_1$, $x_3$) ($x_1$= $x_1$', $x_3$= $x_3$'). Besides, the following can be obtained

$$\begin{cases} x_2' \in [x_1, (x_1 + x_3)/2) & if \;\; x_2 < (x_1 + x_3)/2 \\ x_2' \in [(x_1 + x_3)/2, x_3) & if \;\; x_2 \geq (x_1 + x_3)/2 \end{cases}.$$

(6)

That is, assuming the distance between $x_1$ and $x_3$ (i.e., the length of Q) is $l_Q$, $x_2$ moves less than $l_Q/2$ for embedding $w$.

The process of data extraction and data recovery goes as follows.

Step 1. Calculate the $r$ state of the embedded interval $Q$'($x_1$', $x_2$', $x_3$') with Eq. (1).

Step 2. Divide the embedded interval $Q$' into $2^{c+1}$ equal subintervals, and calculate the length of each subinterval $l_s$ with Eq. (2).

Step 3. Calculate the state value s of the embedded interval Q',

$$s = \lfloor (x_2' - x_1')/l_s \rfloor.$$

(7)

Step 4. Obtain the embedded data $w$,

$$w = s - r \times 2^c.$$

(8)

Step 5. Restore the original coordinate $x_2$,

$$x_2'' = (1 - r) \times x_1' + r \times (x_3' + x_1')/2 + 2^c \times \kappa',$$

(9)

Where

$$\kappa' = x_2' - x_1' - s \times (x_3' - x_1')/2^{c+1}.$$

(10)

The above is Wang and Wang’s approach.

## 3. The Proposed Reversible Scheme

### 3.1 Watermark Embedding Procedure

Let $M$ be the vector map to be watermarked. We embed data into $M$ using the following steps:

Step 1. Scan the vertices of M to get a vertex list V = $\{v_j(x_j, y_j) | j \in \{1,2,\ldots, n\}\}$, where $v_j$ represents the j-th vertex of V, $x_j$ and $y_j$ are the x coordinate and the y coordinate of $v_j$, respectively, and n is the total number of vertices of V.

Step 2. Select two vertices from V as the reference vertices under the control of a private key k. Denote the two reference vertices as $v_{r1}(x_{r1}, y_{r1})$ and $v_{r2}(x_{r2}, y_{r2})$ $(1 \le r1, r2 \le n)$.

Step 3. Calculate the normalized quantization step $Q_w$ according to the Euclidean distance $\|v_{r1}v_{r2}\|$ between $v_{r1}$ and $v_{r2}$ and the vector map's precision tolerance $\tau$,

$$Q_w = \frac{\|v_{r1}v_{r2}\|}{N_w},$$

(11)

Where

$$N_w = \left\lceil \frac{\|v_{r1}v_{r2}\|}{2\tau} \right\rceil.$$

(12)

The parameter $N_w$ obtained here will be directly used as an input parameter in the watermark extraction and data extraction phase.

Step 4. For any vertex $v_j$ ($j \in \{1,2,\ldots, n\}$, $v_j \ne v_{r1}$, $v_j \ne v_{r2}$) of V, embed a watermark $w_j$ ($w_j \in \{0,1,\ldots, 2^c - 1\}$) into it by going through the following 5 parts:

P1. Partition the straight line passing through $v_j$ and $v_{r1}$ into equal segments by the normalized quantization step $Q_w$ starting from $v_{r1}$.

P2. According to the Euclidean distance $\|v_{r1}v_j\|$ between $v_{r1}$ and $v_j$, obtain the index $\tilde{j}$ ($\tilde{j} \in \{0,1,\ldots\}$) of the $\tilde{j}$-th segment $S_{\tilde{j}}$ which $v_j$ is located on,

$$\tilde{j} = \left\lfloor \frac{\|v_{r1}v_j\|}{Q_w} \right\rfloor.$$

(13)

P3. Calculate the two endpoints that define the range of $S_{\tilde{j}}$ $[S^l_{\tilde{j}}, S^r_{\tilde{j}}]$,

$$\begin{cases} S^l_{\tilde{j}} = Q_w \times \tilde{j} \\ S^r_{\tilde{j}} = Q_w \times (\tilde{j} + 1) \end{cases}.$$

(14)

Figure 1 shows the segment which $v_j$ is located on. Since the straight line passing through $v_j$ and $v_{r1}$ is not really exist in the vector map, it is illustrated using a dash line.

P4. Regard $S^l_{\tilde{j}}$, $\|v_{r1}v_j\|$ and $S^r_{\tilde{j}}$ as an interval $Q_j$ ($S^l_{\tilde{j}}$, $\|v_{r1}v_j\|$, $S^r_{\tilde{j}}$), and embed $w_j$ into it using Eqs. (1-5). Denote the watermarked $\|v_{r1}v_j\|$ as $\|v_{r1}v_j{}'\|$.

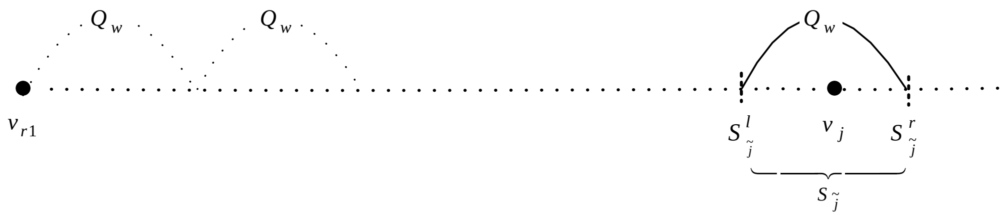

**Figure 1. The Segment which $V_j$ Is Located On**

P5. Move $v_j$ along the straight line passing through $v_j$ and $v_{r1}$ to a new location $v_j'(x_j', y_j')$ so that the Euclidean distance between $v_{r1}$ and $v_j'$ is equal to $\|v_{r1}v_j'\|$,

$$\begin{cases} \dfrac{x_j'-x_{r1}}{x_j - x_{r1}} = \dfrac{\|v_{r1}v_j'\|}{\|v_{r1}v_j\|} \\ \dfrac{y_j'-y_{r1}}{y_j - y_{r1}} = \dfrac{\|v_{r1}v_j'\|}{\|v_{r1}v_j\|} \end{cases} . \tag{15}$$

After embedding data into each vertex, a watermarked vector map $M'$ can be obtained.

In the above procedure, the watermark $w_j$ is embedded into the interval $Q_j(S^l_{\tilde{j}}, \|v_{r1}v_j\|, S^r_{\tilde{j}})$ by moving $\|v_{r1}v_j\|$ between $S^l_{\tilde{j}}$ and $S^r_{\tilde{j}}$ using Eqs. (1-5). According to Eq. (6), the difference between $\|v_{r1}v_j\|$ and $\|v_{r1}v_j'\|$ is less than $Q_w/2$. Since the location of $v_j'$ is obtained by moving $v_j$ along the straight line passing through $v_j$ and $v_{r1}$, the distance between $v_j$ and $v_j'$ is less than $Q_w/2$. According to Eqs. (11-12), we can get that

$$Q_w = \frac{\|v_{r1}v_{r2}\|}{N_w} = \frac{\|v_{r1}v_{r2}\|}{\left\lceil \dfrac{\|v_{r1}v_{r2}\|}{2\tau} \right\rceil} \le 2\tau . \tag{16}$$

Therefore, the distance between $v_j$ and $v_j'$ is no greater than $\tau$, and the validity of the map data can be ensured.

### 3.2 Watermark Extraction and Data Recovery

During watermark extraction and data recovery, similar steps are followed.

Step 1. Scan the vertices of M' to get a vertex list $V' = \{v_j'(x_j', y_j') | j \in \{1,2,\ldots, n\}\}$, where $v_j'$ is the j-th vertex of V', $x_j'$ and $y_j'$ are the x coordinate and the y coordinate of $v_j'$, respectively, and n represents the total number of vertices of V.

Step 2. Select two reference vertices, i.e., $v_{r1}$ and $v_{r2}$ from V' under the control of the private key k.

Step 3. According to $N_w$ and the Euclidean distance $\|v_{r1}v_{r2}\|$ between $v_{r1}$ and $v_{r2}$, calculate the normalized quantization step $Q_w$ using Eq. (11).

Step 4. For any vertex $v_j'$ ($j \in \{1,2,\ldots, n\}$, $v_j' \neq v_{r1}$, $v_j' \neq v_{r2}$) of V', extract the watermark $w_j$ from it and recover the original content of $v_j'$ by going through the following 5 parts:

P1. Partition the straight line passing through $v_j'$ and $v_{r1}$ into equal segments by the normalized quantization step $Q_w$ starting from $v_{r1}$.

P2. According to the Euclidean distance $\|v_{r1}v_j'\|$ between $v_{r1}$ and $v_j'$, obtain the index $\tilde{j}$ of the segment $S_{\tilde{j}}$ which $v_j'$ is located on using Eq. (13).

P3. Calculate the two endpoints that define the range of $S_{\tilde{j}}$ [$S_{\tilde{j}}^l$, $S_{\tilde{j}}^r$] with Eq. (14)

P4. Regard $S_{\tilde{j}}^l$, $\|v_{r1}v_j'\|$ and $S_{\tilde{j}}^r$ as an interval $Q_j'$($S_{\tilde{j}}^l$, $\|v_{r1}v_j'\|$, $S_{\tilde{j}}^r$), and extract the watermark $w_j$ from it using Eqs. (1-2) and Eqs. (7-10), to get the recovered $\|v_{r1}v_j'\|$, *i.e.*, $\|v_{r1}v_j\|$.

P5. Move $v_j'$ along the straight line passing through $v_j'$ and $v_{r1}$ to its original location $v_j$ ($x_j$, $y_j$) using Eq. (15).

After extracting the embedded data from each vertex, the original content of the watermarked vector map $M'$ can be recovered.

## 4. Results and Analysis

We ran experiments on a PC with CPU 2.0 GHz, RAM 3G, Windows 7, Map Objects 2.4 and Microsoft Visual C++6.0. In the experiments, four different 2D vector maps in shapefile format of Environmental Systems Research Institute, Inc. (ESRI) [15] were used as the covers. As shown in Figure2, the four vector maps are a spot heights map of Taylor Rookery [16] (M1), a traffic routes map [17] (M2), a river map [17] (M3) and a lake map of SR41-42 Northern Prince Charles Mountains [18] (M4). Table 1 lists some basic properties of the four vector maps, including the feature type, the number of features/vertices, the scale, the precision tolerance $\tau$ and the density. The term "density" is the average number of vertices within a map patch with the unit area m$^2$. It measures the density of map vertices. Higher density means that the vertices of a map are located closer to each other and their coordinates may have a higher correlation. During embedding, the watermark message used in our experiments was a bit stream generated by using a Gaussian random sequence, and the number of watermark bits each vertex carries $c$ was 1.

Experiment on invisibility: The vector maps in Figure 2 were embedded by the proposed scheme yielding the embedded versions seen in Figure 3. It can be seen that the perceived quality is acceptable.

For evaluating the objective quality of the embedded vector maps, the average distortion $d(M, M')$ and the maximum distortion $Maxd(M, M')$ [14] were calculated,

$$d(M,M') = \frac{1}{n}\sum_{i=1}^{n}\|v_i - v_i'\| ,$$

$$Maxd(M,M') = \max(\|v_i - v_i'\|), \quad (i = 1,2,\ldots,n)$$

(17)

Where $v_i$ and $v_i'$ are the corresponding vertices in the original vector map $M$ and the embedded vector map $M'$, respectively, and $n$ denotes the total number of vertices in the vector map $M$.

Table 2 lists the $Maxd$ and $d$ values of the proposed method and the methods described in [5, 12]. During embedding, the three schemes were performed by taking $c$ = 1. From this table, we can see that because of the configuration of the embedding parameter $Q_w$ using Eq. (11), the $Maxd$ and $d$ values of each vector map do not exceed the precision tolerance $\tau$. The proposed scheme can guarantee the validity of the embedded vector maps. Besides, the invisibility of the proposed method is comparable to that of Wang *et al.*'s method [12]. Wang and Wang's approach [5] cannot guarantee the watermarked vector map quality especially for the 2D vector maps whose coordinates exhibit low correlation, *e.g.*, M2, and M4. That's because Wang and Wang's approach embeds a watermark into a coordinate by moving it between its two neighboring two coordinates in the sorted coordinate list, and the distortion introduced to the coordinate may be great if the distance between its two neighboring two coordinates is not small. Since the proposed method can always guarantee the validity of the embedded vector maps, the invisibility of the proposed scheme is superior to that of Wang and Wang's algorithm.

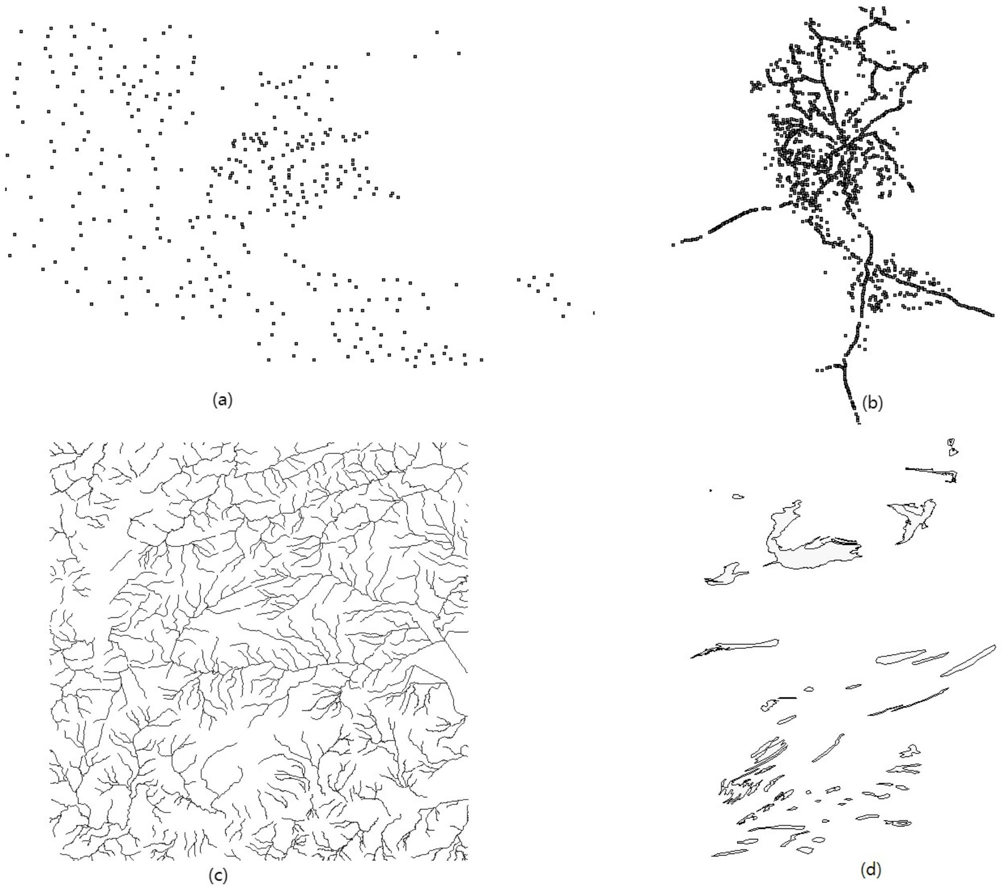

**Figure 2. Test 2D Vector Maps: (a) M1, (b) M2, (c) M3 and (d) M4**

According to the data embedding procedure described in Section 3.1, the invisibility of our proposed scheme is closely related to the number of watermark bits each vertex carries *c*. For embedding *c* watermark bits into a vertex, the vertex is moved within its corresponding segment range using Wang and Wang's method [5]. Since the embedding distortion introduced to a coordinate may be neither monotonic decreasing nor monotonic increasing with the watermark it carries in [5], the embedding distortion may be neither monotonic decreasing nor monotonic increasing with *c* in the proposed method.

**Table 1. Properties of Original Vector Maps**

| Vector maps | Feature type | Features/vertices | Scale | $\tau$ (m) | Density (vertex/m$^2$) |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| M1 | Point | 355/355 | 1:5000 | 0.5 | $1.3014\times10^{-4}$ |
| M2 | Point | 2058/2058 | 1:100000 | 10 | $2.6491\times10^{-8}$ |
| M3 | Polyline | 1084/23854 | 1:25000 | 2.5 | $4.7461\times10^{-4}$ |
| M4 | Polygon | 55/3138 | 1:1000000 | 100 | $5.1888\times10^{-7}$ |

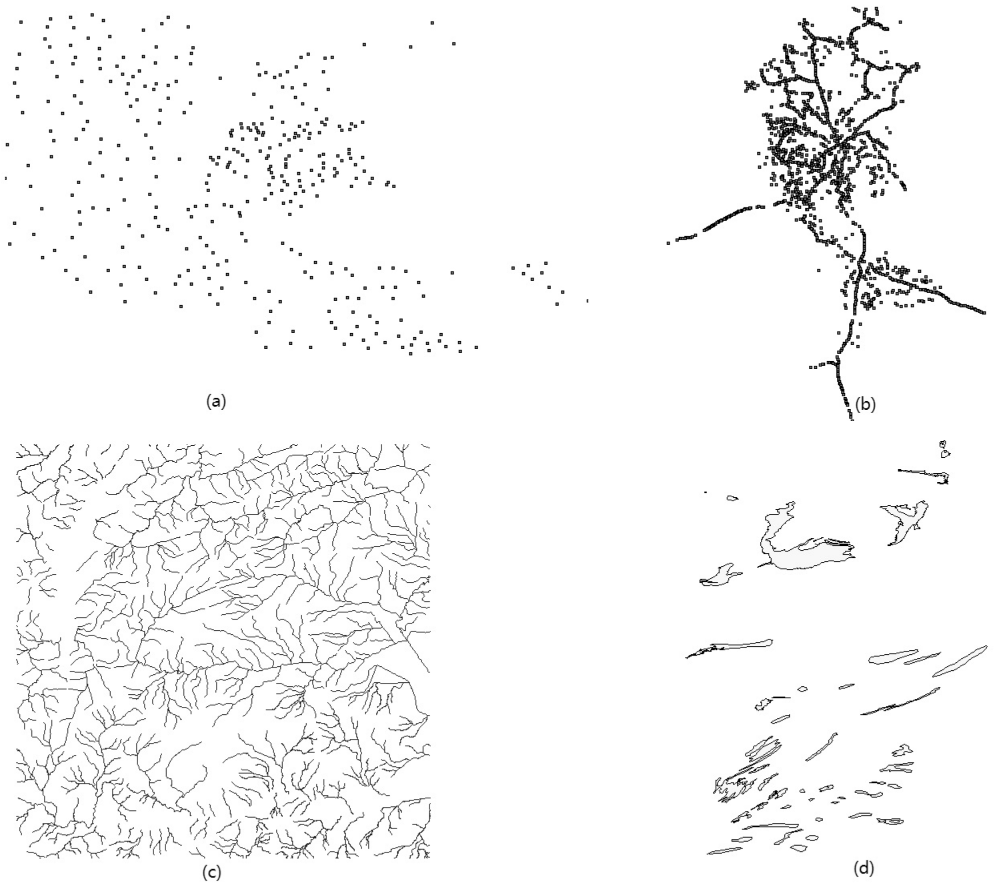

**Figure 3. The Watermarked 2D Vector Maps of Figure 2**

Experiments have been conducted on M1 in Figure 2(a) to demonstrate the relationship between average embedding distortion *d* and *c*. From the experimental results illustrated in Figure 4, we can see that the average embedding distortion is neither monotonic decreasing nor monotonic increasing with *c*. This verifies our analysis above.

Experiment on RST invariance: Experiments have been done to demonstrate the robustness against rotation, uniform scaling and translation transformations of the proposed algorithm. We rotated the watermarked M3 shown in Figure 3(c) by different angles, scaled it with different factors and translated it with different $\Delta x$ and $\Delta y$ in the $x$ and $y$ axes, respectively. From the experimental results shown in Table 3, Table 4 and Table 5, we can see that the BER (bit error rate) of the extracted watermarks is zero. That is, the embedded watermarks can be correctly extracted after RST transformations, and the proposed scheme is invariant to rotation, uniform scaling and translation operations.

**Table 2. The Maxd and D Values of Different Methods**

| Vector maps | Wang *et al*.[12] | | Wand and Wang [5] | | Proposed | |
|---|---|---|---|---|---|---|
| | *Maxd* (m) | *d* (m) | *Maxd* (m) | *d* (m) | *Maxd* (m) | *d* (m) |
| M1 | 0.2381 | 0.1377 | 19.5486 | 1.1246 | 0.2492 | 0.1299 |
| M2 | 4.9334 | 2.7395 | 1672.9117 | 31.8115 | 4.9951 | 2.5040 |
| M3 | 1.2478 | 0.6155 | 1.3866 | 0.0677 | 1.2498 | 0.6266 |
| M4 | 26.1374 | 49.4026 | 514.1794 | 5.5363 | 49.6307 | 24.7573 |

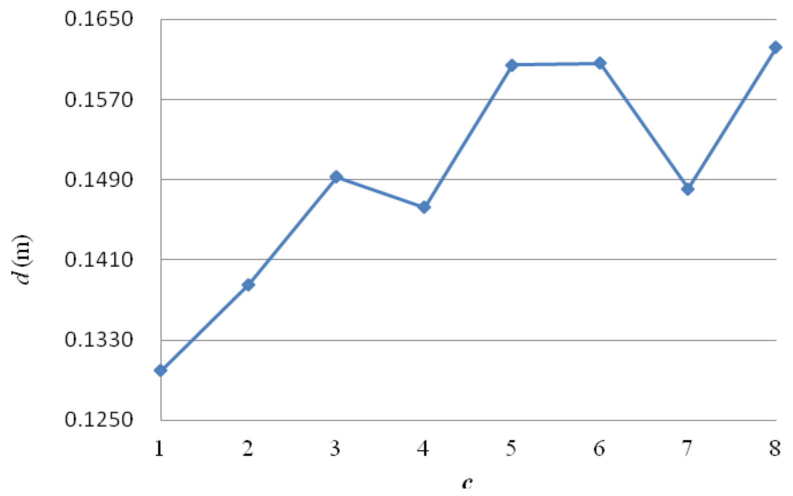

**Figure 4. Relationship between Average Embedding Distortion *d* and *c***

**Table 3. Experiment Results of Rotation**

| Rotation angle (degree) | 30 | 60 | 90 | 150 | 180 | 240 | 300 |
|---|---|---|---|---|---|---|---|
| BER | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

**Table 4. Experiment Results of Uniform Scaling**

| Scale factor | 0.25 | 0.5 | 2.5 | 5.5 | 7.5 | 9.5 |
|---|---|---|---|---|---|---|
| BER | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

**Table 5. Experiment Results of Translation**

| Distance (△x m, △y m) | (– 1.2, 2.3) | (4.2, 5.6) | (2.6, – 7.9) | (– 6.5, – 4.8) |
|---|---|---|---|---|
| BER | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

Experiment on reversibility: To demonstrate the reversibility of the proposed algorithm, the hidden data were extracted from the embedded vector maps in Figure 3 using the proposed method. The *Maxd* and *d* values between the original vector maps and the recovered ones were computed for evaluating the objective quality of the recovered vector maps. Table 6 shows that the *Maxd* values and the *d* values of the four vector maps are all less than $10^{-8}$m. Generally, the storage precision for the coordinates of a 2D vector map is about 0.1mm. In other words, as long as the differences between the original coordinates and the recovered ones are less than $10^{-4}$ m, the data hiding scheme can be seen as reversible. Therefore, the precision requirements of most situations can be met.

**Table 6. The *Maxd* and *D* Values between the Original Vector Maps and the Recovered Ones**

| Vector maps | *Maxd* (m) | *d* (m) |
|---|---|---|
| M1 | $2.8015\times10^{-9}$ | $6.1142\times10^{-10}$ |
| M2 | $2.6494\times10^{-9}$ | $3.3050\times10^{-10}$ |
| M3 | $2.7209\times10^{-9}$ | $3.3401\times10^{-10}$ |
| M4 | $3.5477\times10^{-9}$ | $7.5546\times10^{-10}$ |

Experiment on data capacity: According to the watermark embedding procedure described in Section 3.1, the watermarks can be embedded into nearly all vertices except the vertices that coincide with one of the reference vertices. As a result, nearly every vertex can carry c bits, and the data capacity is about c bit/vertex.

Experiments have been conducted on the vector maps in Figure 2 to compare the data capacity among the proposed scheme and the schemes reported in [5, 12]. During embedding, the three schemes were performed by taking c = 1. The experimental results are listed in Table 7. Since nearly every coordinate can carry c watermark bits in Wang et al.'s scheme [12] (close to 2c bit/vertex), it provides higher data capacity than the proposed scheme. In Wang and Wang's scheme [5], nearly every three adjacent three coordinates in each sorted axis can carry c watermark bits, resulting in c bit/vertex. Hence, the data capacity of the proposed scheme is approximately the same as that of Wang and Wang's scheme [5].

Experiment on computational complexity: Table 8 presents the data embedding and data extraction execution time comparisons among the proposed method and the approaches reported in [5, 12]. During embedding, the three schemes were performed by taking c = 1.

Let's assume n is the number of vertices of the vector map M and $N_F$ be the total number of M's Polylines and Polygons. For Wang et al.'s algorithm [12], the computational complexity is $O(n) + O(N_F \log N_F)$. Because the computational complexity of our proposed approach is O(n), the execution time of the proposed method is lower than that of Wang *et al*.'s method when they are applied to a vector map composed of Polylines/Polygons. For a vector map composed of Points, the execution time of the proposed method and Wang *et al*.'s method are approximately the same, which can be seen from Table 8.

**Table 7. Data Capacity of Different Methods (Bit/Vertex)**

| Vector maps | Wang *et al*.[12] | Wand and Wang [5] | Proposed |
|---|---|---|---|
| M1 | 1.9887 | 0.9972 | 0.9944 |
| M2 | 1.9981 | 0.9558 | 0.9990 |
| M3 | 1.8182 | 0.9770 | 0.9999 |
| M4 | 1.9165 | 0.9758 | 0.9758 |

**Table 8. Data Embedding and Extraction Execution Time of Different Methods (Seconds)**

| Vector | Wang *et al*.[12] | | Wand and Wang [5] | | Proposed | |
|---|---|---|---|---|---|---|
| maps | Embedding | Extraction | Embedding | Extraction | Embedding | Extraction |
| M1 | 0.016 | 0.016 | 0.025 | 0.020 | 0.018 | 0.018 |
| M2 | 0.073 | 0.073 | 0.162 | 0.126 | 0.089 | 0.089 |
| M3 | 3.139 | 3.139 | 3.700 | 2.228 | 2.137 | 2.137 |
| M4 | 0.182 | 0.182 | 0.193 | 0.1595 | 0.125 | 0.125 |

In Wang and Wang's method [5], the main cost which derives from sorting the coordinates during data embedding is $O(n\log n)$. The data embedding execution time of this algorithm is longer than that of the proposed algorithm. Because the coordinates do not need to be sorted during data extraction, the data extraction time of this approach is comparable to that of the proposed algorithm.

## 5. Conclusions

In this paper, we describe a RST invariant reversible watermarking method for 2D vector maps based on the reversible watermarking scheme by Wang and Wang [5]. By dividing the Euclidean distance between a reference vertex and each vertex into equal segments using the normalized quantization step, and embedding data by moving each vertex within its corresponding segment in a revertible manner, the proposed method not only recovers the original content after watermark extraction, but also correctly extracts the embedded watermarks after RST transformations. Besides, the embedding distortions can be controlled by carefully selecting the embedding parameter. Moreover, the proposed method provides low computational complexity, and good reversibility and data capacity.

One drawback of our scheme is that the data capacity is not very high. Our future research will focus on developing RST invariant reversible watermarking schemes for 2D vector maps with high data capacity.

## References

[1] S.-H. Lee and K.-R. Kwon, "Vector watermarking scheme for GIS vector map management", Multimedia Tools and Applications, vol. 63, no. 3, **(2013)**.

[2] J. Lafaye, J. Béguec, D. Gross-Amblard and A. Ruas, "Blind and squaring-resistant watermarking of vectorial building layers", Geoinformatica, vol. 16, no. 2, **(2012)**.

[3] V. R. Doncel, N. Nikolaidis and I. Pitas, "An optimal detector structure for the Fourier descriptors domain watermarking of 2D vector graphics", IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 5, **(2007)**.

[4] F. Peng, Y. Liu and M. Long, "Reversible watermarking for 2D CAD engineering graphics based on improved histogram shifting", Computer-Aided Design, vol. 49, **(2014)**.

[5] P.-C. Wang and C.-M. Wang, "Reversible data hiding for point-sampled geometry", Journal of Information Science and Engineering, vol. 23, no. 6, **(2007)**.

[6] J. Tian, "Reversible data embedding using a difference expansion", IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 8, **(2003)**.

[7] M. Voigt, B. Yang and C. Busch, "Reversible watermarking of 2D-vector data", Proceedings of the Multimedia and Security Workshop, **(2004)**.

[8] X. Wang, C. Shao, X. Xu and X. Niu, "Reversible data-hiding scheme for 2-D vector maps based on difference expansion", IEEE Transactions on Information Forensics and Security, vol. 2, no. 3, **(2007)**.

[9] L. Zhou, Y.-J. Hu and H.-F. Zengm, "Reversible data hiding algorithm for vector digital maps", Journal of Computer Applications, vol. 29, no. 4, **(2009)**.

[10] L. Cao, C. Men and R. Ji, "Nonlinear scrambling-based reversible watermarking for 2D-vector maps", The Visual Computer, vol. 29, no. 3, **(2013)**.

[11] L. Cao, C. Men and Y. Gao, "A recursive embedding algorithm towards lossless 2D vector map watermarking", Digital Signal Processing, vol. 23, no.3, **(2013)**.

[12] N. Wang, H. Zhang and C. Men, "A high capacity reversible data hiding method for 2D vector maps based on virtual coordinates", Computer-Aided Design, vol. 47, **(2014)**.

[13] N. Wang and C. Men, "Reversible fragile watermarking for 2-D vector map authentication with localization", Computer-Aided Design, vol. 44, no. 4, **(2012)**.

[14] N. Wang and C. Men, "Reversible fragile watermarking for locating tampered blocks in 2D vector maps", Multimedia Tools and Applications, vol. 67, no. 3, **(2013)**.

[15] ESRI shapefile technical description. http://www.esri.com/library/whitepapers

[16] U. Harris, "Taylor Rookery 1:5000 Topographic GIS Dataset Australian Antarctic Data Centre-CAASM Metadata", https://data.aad.gov.au/aadc/metadata/metadata_redirect.cfm?md=/AMD/AU/Tayl5k.

[17] http://www.ibge.gov.br/english/geociencias/default_prod.shtm.

[18] U. Harris, "SR41-42 Northern Prince Charles Mountains - 1:1 Million Topographic GIS Dataset", Australian Antarctic Data Centre - CAASM Metadata. http://gcmd.nasa.gov/

KeywordSearch/Metadata.do?Portal=amd_au&MetadataView=Full&MetadataType=0&KeywordPath =&OrigMetadataNode=AADC&EntryId=SR41-42.