




# Secure map legends based on just noticeable distortion and watermark bit recovery

Lin Zhou<sup>1</sup> · Xiaohui Yuan<sup>2</sup>  · Yuanyuan Liu<sup>1</sup> · Zhanlong Chen<sup>1</sup> · Peng Xie<sup>1</sup>

Received: 8 June 2021 / Revised: 22 April 2022 / Accepted: 31 January 2023 /

Published online: 8 March 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

## Abstract

An important component of cartography, the map legends can be a type of hybrid data with small data quantity and high precision. Therefore, its watermark method should be versatile to all map legend data types. In addition, its watermark should be robust, imperceivable, and small. In this regard, our study proposes a novel watermarking method of map legends for copyright protection. The data types of map legends are evaluated. According to the data types and just noticeable distortion tolerance, the watermark bit embedding positions of each data type are defined. A text watermark is adopted to reduce the number of watermark bits. By the watermark bit count, map legends are divided into groups to contain multiple watermarks. Meanwhile, a watermark bit recovery method is designed to fix the damaged watermark bits based on the extracted ones, which ensures greater robustness. The experimental results demonstrate the proposed method achieves the expected goal in terms of various types of attacks and image operations.

**Keywords** Security · Legends · Watermarking · Copyright protection

## 1 Introduction

The map legends are graphical representations of information, with design principles similar to a map, and are known as the language of maps. Creating map legends that are effective to the map is time-consuming and tedious work that must follow many layout concepts such as visual balance, clarity, logical groupings, and unambiguous references [19]. Well laid out legends enhance the gestalt of the maps. To avoid repeated work and achieve consistency, techniques for map legend sharing have been developed [16]. However, It is important to protect the copyright of map legends to facilitate commercial and restricted usage of such intellectual property [14, 18].

Digital watermarking embeds a marker (watermark) in noise-tolerant digital works such as images, videos, etc. [2, 7, 9, 21] The watermark helps verify the authenticity or

---

✉ Xiaohui Yuan  
xiaohui.yuan@unt.edu

<sup>1</sup> China University of Geoscience, Wuhan, China

<sup>2</sup> University of North Texas, Denton, Texas, USA

integrity of the works and identify the owners of the works. It may also help to trace copyright infringements and for banknote authentication [4, 6]. In recent years, digital watermarking methods of geographic information have been studied deeply, especially digital watermarking for the vector and raster geographic data [1, 11, 13].

Digital watermarking technology can be applied to protect the copyright of map legends. However, map legends are different from the conventional imagery and multimedia data [8]. They are stored and managed by map legend libraries. A map legend library or a map legend may contain one or more data types of vector graphics, text, images, etc. Unlike many contemporary multimedia files, map legend libraries and map legends usually have relatively small size [1]. Last, map legends must be highly accurate because their precision directly affects the accuracy of maps [10]. Due to these characteristics, the watermarking method for map legends should have small capacity requirements, low disturbance, and be suitable for all data types.

To protect the accuracy of map legends, Zhou et al. [20] proposed a zero-watermarking method. Zero-watermarking extracts important and robust features of protected data to construct a watermark without modifying the data [15]. The zero-watermarking method [20] takes item amount, item types of maximum and minimum items of item elements, and the element type with the least number of each map legend as key characteristics and combines them with the watermark bits to generate a zero-watermark outside rather than embed watermark bits into these map legends. In this process, the zero-watermarking method makes no change to map legend data and there is no watermark capacity limitation and data perturbation [20]. However, the watermarks can be extracted from entities that are not protected due to pseudo-validation. This is due to the high similarity of key characteristics extracted from the entities. Map legends are often specified by the national cartographic specifications. When making map legends, the vendors must strictly comply with these specifications so that their map legends for the same scale maps are uniform in color, size, shape, etc. It is difficult to find out distinguishable key characteristics among them [5, 12]. Therefore, false verification is inevitable when using zero-watermarks to protect map legends.

To overcome pseudo-validation and improve the robustness of watermarks for map legends, this paper proposes a watermark embedding method for copyright protection. Our contributions include the following:

1. Design a Watermark Bit Recovery Method (WBRM) to improve the robustness of the embedded watermark. Multiple watermark copies are embedded in the host data and WBRM recovers a watermark based on the other watermark copies.
2. Adopt text watermark instead of image watermark to reduce the number of watermark bits. Compared with image watermark, text watermark needs fewer bits, which makes it possible to embed more watermark copies in a map legend library, thus improving the robustness of the watermark. In addition, it reduces the capacity requirement of map legends.
3. Define watermark bits embedding position based on Just Noticeable Distortion (JND) to make the embedded watermarks imperceptible. Hence, it invites fewer intentional attacks.

The rest of this paper is organized as follows. Section 2 reviews the data structure of map legends, Arnold transformation, and Unicode used in watermarking. Section 3 describes the proposed watermarking method in detail including preprocessing steps, legend grouping, and watermark extraction algorithm. Section 4 discusses the experimental results with respect to various attacks and comparison studies with image watermarks and existing methods. Section 5 concludes this paper with a summary.

## 2 Background

### 2.1 Data structure of map legends

The real entities are abstracted into point features, line features, and area features, which are represented with point legends, line legends, and area legends. Although the data structures of map legends vary among vendors, most of them take legend items as basic units. A map legend usually consists of a legend header and legend items. The legend header defines the basic information of a map legend such as legend ID, name, anchor point/line, item amount, etc. In a map legend library where map legends are stored, each legend ID is unique. An anchor point of a point legend indicates the location where the legend should overlap with the coordinates of a point feature. An anchor line of a line legend indicates the location where the legend should overlap on the baseline of a line feature. The setting of an anchor point/line is related to the shape of a map legend. Item count records the number of legend items contained by a map legend. A map legend must have at least one legend item. Legend items can have attributes and figure data or only have attributes, which define how to render the graphics. Item data can be coordinates, image pixels, or shape parameters such as the center and radius of a circle. Table 1 summarizes the attributes of legend types including their categories and sub-categories.

Point legends have four item types: primitive geometric item, Truetype, image, and point legend reference. Primitive geometric item refers to polyline, curve, polygon, rectangle, filled rectangle, circle, filled circle, ellipse, filled ellipse, and arc. Truetype item is stored in the form of Truetype font. Image item can be an image in BMP, TIFF, JPG, or PNG format. Point legend reference item takes the other point legend in the same library as an item. It only includes the ID and attributes of the referred legend rather than its specific data.

Line legends include primitive line, point legend references, and line legend references. The primitive line legend item is viewed differently. In one case, the primitive line legend items include solid line, dashed line, dashed-dotted line, double dashed line, double solid line, dotted line, anchor point, traverse line, traverse point, zigzag line, gradient width solid

**Table 1** Legend types and attributes. Sub-Categories are in parenthesis

Legend	Category	Attribute
Point	geometric item (polyline, polygon, rectangle, curve, arc, rectangle (filled), ellipse (filled), circle (filled))	color, width, cap, joint
	True type	type, color, angle, anchor point, size
	Image	type, angle, anchor point, size
	Point legend reference	angle, color, anchor point, size, referred legend ID
Line	primitive line (solid, dashed, dotted)	color, width, cap, joint
	primitive line (program line)	color, width, program ID
	point legend reference	angle, color, anchor point, size, referred legend ID
	line legend reference	color, anchor line, width, referred legend ID
Area	color (single)	color
	color (gradient)	gradient direction, start point, color1, color2
	Point legend reference	angle, color, size, referred legend ID
	Line legend reference	color, line-width, baseline, referred legend ID

line, gradient width dashed line, ribbon halo line, etc. In another case, they are the original lines provided by a drawing environment including solid lines, dashed lines, dotted lines, dashed-dotted lines, etc. The third view puts forward that some line legends are unable to break into repetitive units such as electrical line legend, gradient single line legend, etc. And the number of these legends is small, so the best expression of them is the program line which is real-time rendered by a program. Based on these views, we consider that the primitive line legend items include solid line, dashed line, dotted line, and program line. The line legend reference item is similar to the point legend reference item. The difference only lies in that a line legend reference item is associated with a line legend in the same library.

Area legends have three item types: color, point legend reference, and line legend reference. The color item can be a single color or gradient color. The former can express water area, administrative division, etc. The latter is usually used to describe quantity change such as different water depths in a water area.

## 2.2 Arnold transform

Arnold Transform enhances the security of the watermark. In mathematics, Arnold transformation, also called cat map transform, is a chaotic map from the torus into itself [3]. It is the discrete analog of the cat map transform that is widely used in digital image watermarking. The discrete Arnold transform used to watermark images is usually defined as:

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} 1 & b \\ a & ab + 1 \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} \bmod N, \quad 0 \leq i, j \leq N - 1 \quad (1)$$

where  $a, b$  are integers,  $N \times N$  gives the size of the image,  $(i, j)$  denote the coordinates and  $(i', j')$  is its coordinate after Arnold transform. In the next iteration, the output of the left part  $(i', j')$  is the right input. This process is repeated until the image becomes chaotic and is unable to be identified, i.e., a scrambled image. Different  $a$  and  $b$  lead to different periodicity, which makes  $a, b$  an option for the key. Because of the periodicity of Arnold transform, an image watermark that is randomized by the transformation can be restored to its original state. On the other hand, decryption is infeasible without the scrambling algorithm and the key.

## 2.3 Unicode

Unicode has been used in the internationalization and localization of computer software. The commonly used encodings are UTF-8 and UTF-16. They are both variable-length character encodings. Depending on the number of significant bits in the numerical value of the code point, UTF-8 encodes code points in one to four bytes. For the first 128 code points, UTF-8 uses one byte with the highest bit '0'. Since these Unicode code points are ASCII characters, an ASCII text is also a UTF-8 text. From two bytes to four bytes, the highest byte starts with consecutive '1's to denote the number of bytes, and the other bytes all start with '10's.

UTF-16 is capable of encoding all valid code points of Unicode with two or four bytes. Code points from U+0000 to U+FFFF, UTF-16 encodes them by two bytes that are numerically equal to the corresponding code point. Code points from U+010000 to U+10FFFF are encoded as four bytes. First, 0x10000 is subtracted from the code point, leaving a 20-bit number in the range 0x00000–0xFFFFF. The higher ten bits are added to 0xD800, and the lower ten bits are added to 0xDC00.

### 3 Proposed method

#### 3.1 Embedding position

Once watermark bits are embedded in map legends, the data of map legends have to be changed. For map legends, the most basic changeable factors, which cause perceivable differences, are called visual variables. The generally accepted visual variables include color, size, orientation, and shape. Color heightens the aesthetic perception of maps. On maps, color expresses qualitative and quantitative characteristics of geographical elements. The hue, brightness, and saturation of colors have different effects on maps, called color-dependent variables. A map legend can contain one or more colors. Legend size is often used to express the geographical features of quantitative characteristics such as length, width, height, area, volume, etc. In map legend design, the different qualitative characteristics of the geographical features can be expressed with the orientation change of legends. The meanings of orientation include the direction change of the entire legend graphics, which is common to point legends, and the direction change of textures, which is common to line and area legends. The shape is the form of map legend or its external boundary, outline, or external surface, as opposed to other properties such as color, texture, or material composition. Legend shape is often defined by the shape and assembly of its items.

To make visual variables change imperceptibly after watermark bits are embedded into the map legends, it is essential to find out the best watermark bit embedding position. Humans can identify a single line with 0.02mm to 0.03 mm thick. The smallest, identifiable geometric figures are associated with their structures and complexity. When edge length is equal to or greater than 0.3 mm, a solid rectangle can be identified. To hollow rectangles, their edge length should be 0.4 mm at least. Usually, a complex geometric figure needs a larger size than the simple one to be identified by humans. Although the RGB color model has  $2^{24}$  colors in total, a common individual can only distinguish about 1 million of them. Among the three primaries R, G, and B, the human eyes are the least sensitive to blue.

Based on these observations, we define the watermark embedding positions accordingly. For a raster item or TrueType item, its size is modified to hide a watermark bit. For a primitive geometric item with lines, its line width is modified to hide a watermark bit. Otherwise, its color is modified to hide a watermark bit. For a reference item without a color attribute, its size is changed to hide a bit.

#### 3.2 Watermark embedding

The proposed embedding method is illustrated in Fig. 1. The watermark is in text form and the preprocessing of this text includes three steps: 1) translate each character of the watermark into Unicode, 2) encode these code points using UTF-8 or UTF-16, 3) the code sequence is transformed into a square matrix by padding with '0' characters. After preprocessing, the text watermark is transformed into a binary image matrix for the application of the Arnold transformation. The watermark preprocessing process is summarized in Algorithm 1.

Map legends also need to be preprocessed. All legends in a library are classified into point legend sets, line legend sets, and area legend set. The three sets are further divided into groups according to the size of the encrypted watermark respectively. Each group is embedded in a watermarked copy. Grouping helps find out the attacked watermark copies in the process of watermark extraction. Suppose a legend set has  $T$  legends in all, and a record file, denoted with  $F$ , is created. Our grouping method is presented in Algorithm 2.

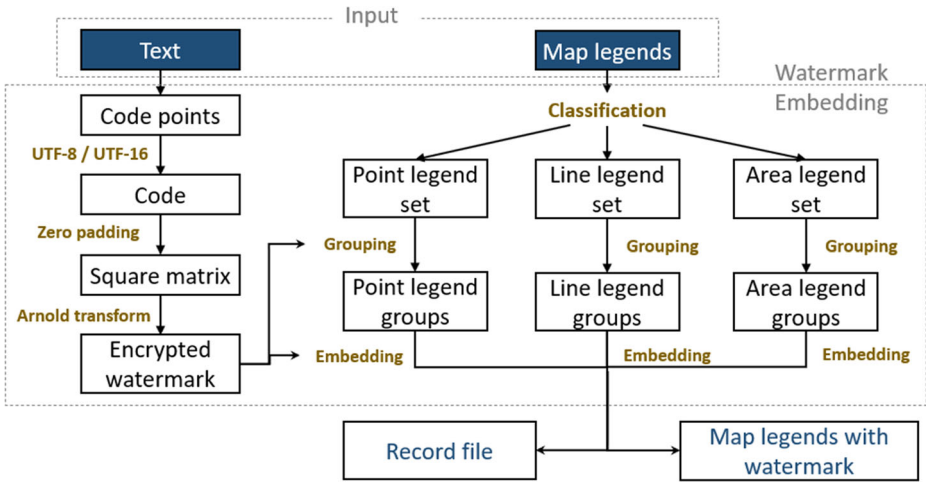


Fig. 1 The flowchart of watermark embedding

Input: text watermark

1. Watermark characters → Unicode code points
2. Unicode code points → UTF-8 or UTF-16 codes sequence S
3. Calculate sequence length  $L = \text{length}(S)$
4. Calculate  $k = \sqrt{L}$
5. Padding  $(k \times k - l)$  0 characters to S
6. Transform S into a square matrix
7. Perform Arnold transform to M
8. Turn the result of the last step into a 1-D sequence W

**Algorithm 1** Watermark preprocessing.

Input: a legend set,  $K = k \times k, c = 0, j = 0$   
 Initial a new group  
**while**  $T > 0$  **do**  
     Get the number of items of the first legend and add it to  $c: c = c + i$   
     Add the legend to the group  
     Remove the legend from the set  
      $T = T - 1$   
     Record legend IDs of the group to a file named F  
     **if**  $c \geq K$  **then**  
         Record legend IDs of the group to F  
         Make a new group and set  $c = 0$ .  
     **end if**  
**end while**

**Algorithm 2** Legend grouping.

```

Input: map legend library and i
Output: map legends with watermark and record file F
for each legend group do
     $i = 0$ 
    for each legend do
        for each legend item do
            if  $i < K$  then
                Embed a watermark bit into it
                 $i = i + 1$ 
            end if
        end for
        Record the start and end bit indexes of W to F
    end for
end for
    
```

**Algorithm 3** Watermark embedding.

Although the last group may not have enough items to contain a whole watermark, the incomplete watermark bits will contribute to the recovery of the watermark. In addition, every group can contain an integrated watermark. For each legend item of a legend in a group, one watermark bit will be embedded. Like the previous analysis, the embedding position of each item depends on its item type. If the watermark bit is ‘1’, add 1 to the tail of the attribute value; otherwise, keep it unchanged. The details of our embedding process are described in Algorithm 3. The outcome of the watermark embedding includes map legends with the watermark and the record file. In legend grouping and watermark embedding, the record file contains information about legend groups and watermark bits distribution. An example of the record file is shown in Table 2.

### 3.3 Watermark extraction

Figure 2 illustrates the flowchart of our watermark extraction method. The input to our method includes the original map legends, the record file, and the secret key a, b for Arnold transform. Hence, it is a non-blind watermarking method.

**Table 2** An example of partial record file

Content	Example
Legend Type: point legend	PT
Legend Group ID 1	PT00000001
Legend ID start bit index end bit index	000000001 0 4
...	
Legend Group ID 2	PT00000002
Legend ID start bit index end bit index	00000000n 0 3
...	
Legend Type: line legend	LN
...	
Legend Type: area legendlegend	AA
...	

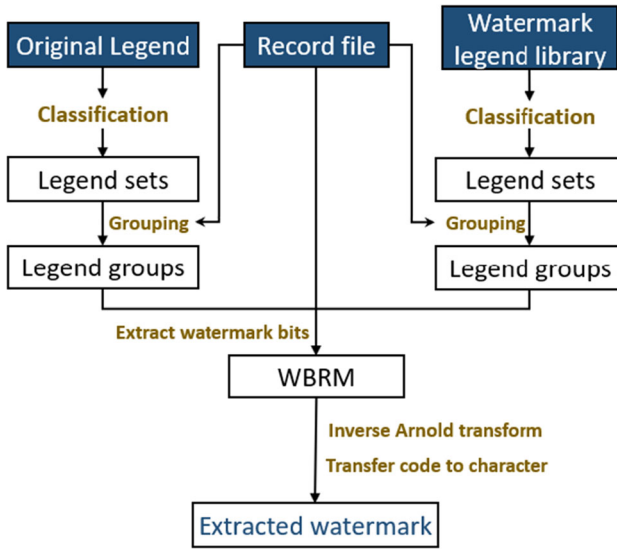


Fig. 2 The flowchart of watermark extraction

Like the watermark embedding process, the original map legend library  $A$  and the watermarked map legend library  $A'$  are firstly classified into three legends sets. Then, each set is further divided into legend groups according to information of legend groups recorded in  $F$ . Take a point legend group recorded in  $F$  as an example, given it consists of four legends with legend IDs ID1, ID2, ID3, and ID4. If  $A$  or  $A'$  has four legends with the above IDs, the four legends will be taken as a group. If a legend among them is not existing, it will be labeled and the remains will still be taken as a group. Next, watermark bits of each group are extracted, and WBRM is operated if needed. Let  $SA$  denote a legend in  $A$ ,  $SA'$  denotes the corresponding legend of  $SA$  in  $A'$ ,  $v(i, A)$  is the attribute value of the  $i$ th item in  $SA$ ,  $v(i, A')$  is the attribute value of the  $i$ th item in  $SA'$ , and  $W_i$  represents the  $i$ th extracted watermark. The process is presented by Algorithm 4. Afterward,  $W'$  is turned into a 2-D binary array  $M'$ , and  $M'$  is inversely Arnold transformed by the secret key  $a$  and  $b$ . Last,  $M'$  is converted into a binary stream  $S'$ , and  $S'$  is turned into code points by the Unicode encoding. By the code points, get the corresponding characters that are the recovered watermark.

## 4 Experimental results

### 4.1 Settings

To evaluate the performance of the proposed watermarking method, experiments are conducted on map legends of a legend library named “25w” in MapGIS K9. The library contains 752 point legends, 256 line legends, and 66 area legends. These legends are divided into groups based on the number of watermark bits. Watermarks need to be designed and preprocessed.

A text watermark is shown in Fig. 3(a). It contains letters and numbers. Using UTF-8 to encode this text watermark, we have the encoded matrix as shown in Fig. 3(b) following



---

Input: map legend library A and A', SA, SA',  $v(i, A)$ ,  $v(i, A')$ ,  $W_i$   
Output: Recovered watermark  $W'$

```

for each legend group do
  Get legends from A' by the legend IDs recorded in F
  if there are non-existent legends then
    Mark their corresponding watermark bits as x characters
  else
    Get their corresponding legends from A.
    if SA and SA' have different numbers of items then
      Mark its corresponding watermark bits as x characters
    else
      for each  $v(i, A)$  and  $v(i, A')$  do
        Round  $v(i, A)$  and  $v(i, A')$  to the same precision
        Take the last number of them to do subtraction
        if the absolute difference is greater than 1 then
          Set the watermark bit as x.
        else
          Take it as a watermark bit
        end if
      end for
    end if
  end for
if there is a  $W_i$  without x then
   $W' = W_i$ 
else
  Find out the  $W_i$  with the least x among all the recovered watermarks,
   $W' \leftarrow W_i$ 
  for each x in  $W'$  do
    Use its index to retrieve the remaining  $W_i$ 
    if there is a bit with '0' or '1' value in  $W_i$  then
      Use the value to replace the corresponding x in  $W'$ 
    else
      Move to next x in  $W'$ 
    end if
  end for
end if

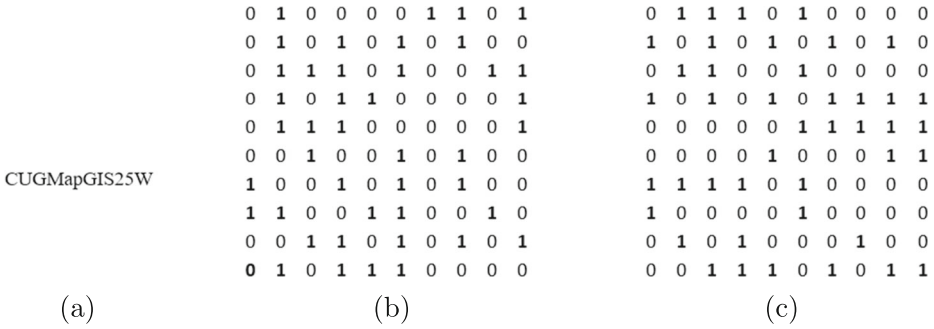
```

---

**Algorithm 4** Extracting and WBCM.

Algorithm 1. The cycle of Arnold transform is proportional to its size. Figure 3(c) shows the result of the Arnold transform.

Since legends are comprised of different basic items, the number of groups is not proportional to the number of legends. Based on the number of watermark bits, point legends are divided into 104 groups, line legends are divided into 28 groups, and area legends are divided into 35 groups. The number of legend groups decided the times that the watermark can be embedded. That means the library can embed a watermark 103 times in point legends, 27 times in line legends, and 34 times in area legends.



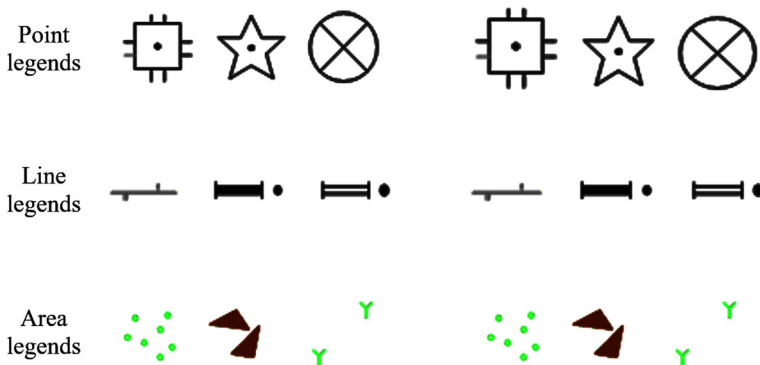
**Fig. 3** (a) text watermark. (b) watermark matrix encoded with UTF-8. (c) result of Arnold transform (repeated eight times and a=3, b=5)

### 4.2 Visibility and robustness

The evaluation factors of digital watermarking algorithms mainly include robustness, watermark capacity, and visibility. As embedding watermarks depends on the content of the host data, the capacity of a watermark is closely related to the amount of host data. If the watermark capacity is not large enough, it is unlikely to embed the watermark in the host. Otherwise, if its capacity is large, the host data can not only embed one complete watermark but even embed multiple watermark copies, which increases the robustness under intentional attacks.

Based on the human visual system or human audio system and watermarking algorithms, the embedded watermarks should be imperceptible to human senses and the host data show no obvious degradation after embedding watermark information. In our evaluation of watermark visibility, watermarks are embedded in all three types of legends. We compare the map legends that contain watermark bits with the originals. Figure 4 shows nine randomly selected map legends with and without watermark bits. It is clear that there is no visual difference between the legends with and without watermarks. Hence, the inserted watermark bits are imperceptible to human beings.

Watermark robustness refers to the ability to successfully extract watermarks hidden in host data after attacks. The robustness of a watermarking algorithm is poor if the watermark



**Fig. 4** Visibility of watermarks. Original legends (left) and legends with watermark bits (right)

**Table 3** The extracted watermarks after moving attacks

Removed	Point legends	Line legends	Area legends
30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W

The watermark text is ‘CUGMapGIS25W’

cannot be extracted correctly after an attack; otherwise, the algorithm is robust. The watermark attack of map legends mainly includes legend moving, legend adding, legend editing, and legend deletion.

1. *Moving attacks*: Legend moving is to change the positions of legends in a legend library. It includes position changes of partial legends and all legends. In this experiment, legends are randomly moved by 30%, 45%, 60%, 75%, and 90%. The results of watermark extraction are shown in Table 3. The watermarks embedded in the images are the same text ‘CUGMapGIS25W’. The movement of the legend makes no changes to the watermark category and ID, which induces no distortion to our method. As shown in this table, the extracted watermarks from the attacked legends match the embedded texts for all three types of map legends. It is hence evident that the moving attacks induce little, if any, damage to the watermarks.
2. *Adding attacks*: Legend adding attack is adding some legends without watermark bits to a legend library with watermarks. Legends may be added to anywhere of the legend libraries. To simulate this attack, some legends are randomly removed from the legend library, and the remains of each legend type should be able to contain a whole watermark at least. Then, the watermark is embedded and the removed are returned as added legends. We then extract the watermarks. The test is performed three times and the results are reported in Table 4. The recovered watermarks match the embedded one in all cases. It is clear that adding attacks cause no damage to the watermarks.
3. *Editing attacks*: Legend editing includes a series of operations such as item addition, deletion, data changing, attribute changing, etc. Not all operations affect the watermark extraction. Only the editing attacks that change the data with watermark bits can interfere with the embedded watermarks. To get an understanding of the robustness of editing attacks, all editing attacks are performed on the data with hidden bits. Average editing, random editing, and mass editing are evaluated. Average editing attacks alter legends with uniform intervals; random editing attacks randomly change legends;

**Table 4** The extracted watermarks after adding attacks

Repetition	Point legends	Line legends	Area legends
1	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
2	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
3	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W

The watermark text is ‘CUGMapGIS25W’

**Table 5** The extracted watermarks after editing attacks

Edit		Point legend	Line legend	Area legend
Average editing	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	!U{BEL}E.P{HT}IQ25V	{EXT}TGMaPIS25{VT}
Random editing	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	3{BS}L+aZGK{EXT}2> W	{STX}UGMaPIS25{ESC}
Mass editing	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	?{NAK}(MapGI{DC1}25T	CUGMapGIS25W

The watermark text is ‘CUGMapGIS25W’. There exist invisible characters such as the ‘new line’ character, which are represented with curly brackets, e.g., {LF}

mass editing attacks modify successive legends. The percentages of the edits are 30%, 45%, 60%, 75%, 90%, and 95%. The results of watermark extraction from map legends after editing attacks are shown in Table 5. In all cases, the embedded watermarks are ‘CUGMapGIS25W’.

As the amount of edits increases from 30% to 90%, our method successfully recovers the embedded watermarks from the three categories of map legends. When we increase the amount to 95%, which is a severe change to the map legends, we observe incorrect watermark extraction from line and area legends. The incorrect ones are highlighted in blue in Table 5. However, the extracted watermarks from point legends at 95% edits remain intact, which is attributed to the relatively large number of point legends.

4. *Deletion attacks*: In this experiment, some map legends are removed after watermark embedding. We evaluate our method with average deletion, random deletion, and mass deletion. The average deletion removes legends with a uniform interval, the random deletion removes legends randomly, and the mass deletion removes successive legends. The percentages of deleted legends are 30%, 45%, 60%, 75%, 90%, and 95%. The results of watermark extraction are shown in Table 6. Similar to the editing attacks, the deletion attacks impose no impact on the integrity of the embedded watermark when the rate of deletion increases up to 90%. The watermarks are successfully recovered. When the amount of deletion is at 95%, we observe incorrect recovered watermarks for both line and area legends.

Based on our experimental results of various attacks, we can conclude that the legend moving and legend adding attacks impose no impact on the embedded watermarks using

**Table 6** The extracted watermarks after deletion attacks

Deletion		Point legend	Line legend	Area legend
Average deletion	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	SUfJ/pYIq25k	WuGMap]]S25z
Random deletion	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	[[{LF}G^aQ{ACK}IS''5{ETB}	CUG{DC3}ap-6Sr5W
Mass deletion	30%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	45%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	60%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	75%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	90%	CUGMapGIS25W	CUGMapGIS25W	CUGMapGIS25W
	95%	CUGMapGIS25W	{DEL}5+MarGI325W	CUGMapGIS25W

The watermark text is ‘CUGMapGIS25W’. There exist invisible characters such as the ‘new line’ character, which are represented with curly brackets, e.g., {LF}

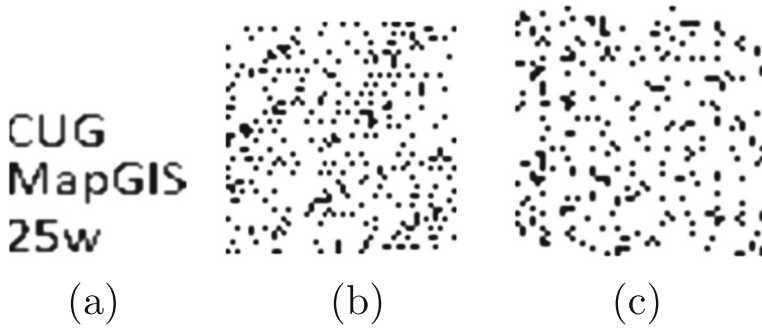
our proposed method. Because legend ID and watermark bit indexes of each map legend are recorded, it is easy to rebuild the order of map legends and, hence, ignore the new legends. Intact watermarks are also successfully extracted under legend editing and legend deletion attacks even when the amount of changes reaches 90%. With the increasing volume of legends edited or deleted, more operations of WBRM are usually needed. The damage to the watermark is related to the number of legend groups in each category. We observe that fewer legend groups lead to greater damage. It is clear that the proposed method is robust to the common attacks to map legends.

### 4.3 Comparison with image watermarks

As images are often used as a watermark, we compare the image watermark and text watermark. Figure 5(a) shows an image watermark that has the same content as the text watermark. This image watermark is monochrome and its text is in 6 pts. Figure 5(b) and (c) illustrate the scrambled images after 7 and 19 times, respectively.

The size of the image watermark is 50 by 50, which is more than 25 times the text watermark. Table 7 lists the number of image and text watermarks embedded in the test map legend library.

Depending on the volume of the legend category, the number of embedded image watermarks and text watermarks varies. However, in all cases, we can embed more text watermarks to the map legends. That is, text watermarks require a smaller amount of space for proper embedding. The feasibility of embedding multiple copies of watermarks makes



**Fig. 5** (a) image watermark. (b) scrambled image (7 times). (c) scrambled image (19 times)

the use of text watermarks more robust than the use of image watermarks. The legend library in our experiments belongs to the topographic map legend library. There are many thematic map legend libraries in GIS applications, many of which usually contain few legends. Large watermarks such as images face the challenge of insufficient space of the host data for proper embedding. This is clearly an advantage for small watermarks such as text used in our method.

As we demonstrated that legend moving and legend adding attacks impose no impact on the embedded watermarks processed by our proposed method, both image watermarks, and text watermarks can have the same robustness under such attacks. We discuss the security of text watermarks and image watermarks under legend editing and legend deletion attacks.

1. *Editing attacks*: In this experiment, we follow the same process of simulating the legend editing attacks as described in Section 4.2 and embed image watermarks on the same legend library. Average editing, random editing, and mass editing are evaluated. The percentages of the edits are 30%, 45%, 60%, 75%, and 90%. Table 8 gives the recovered image watermarks for point, line, and area legends.

As the percentage of the edits increases, we observe degradation of the recovered watermarks. Among the three legend categories, point legends suffer the least degradation in all cases. Both line and area legends exhibit poorly recovered watermarks starting at 30% of editing attacks. However, when the edits reach 90%, the recovered watermarks in all cases are scrambled. In comparison to our evaluation of text watermarks (Section 4.2), image watermarks are less robust to editing attacks.

2. *Deletion attacks*: Following the experiments presented in Section 4.2, we evaluate image watermarks under average deletion, random deletion, and mass deletion are performed and evaluated. After each type of deletion attack, the extracted watermarks are shown in Table 9. We observe very similar patterns to the results of editing attacks. Since the experimental data are the same, the results of the text watermark reported








**Table 7** The number of embedded image and text watermarks

Watermark	Point legend	Line legend	Area legend
Image	4	1	1
Text	103	27	34

**Table 8** The extracted image watermark after editing attacks

Edit		Point legend	Line legend	Area legend
Average deletion	30%	CUG MapGIS 25w		
	45%	CUG MapGIS 25w		
	60%	CUG MapGIS 25w		
	75%	CUG MapGIS 25w		
	90%			
Random deletion	30%	CUG MapGIS 25w		
	45%	CUG MapGIS 25w		
	60%	CUG MapGIS 25w		
	75%	CUG MapGIS 25w		
	90%			
	30%	CUG MapGIS 25w		
	45%	CUG MapGIS 25w		

**Table 8** (continued)

Edit		Point legend	Line legend	Area legend
Mass deletion	60%	CUG MapGIS 25w		
	75%	CUG MapGIS 25w		
	90%			

in Section 4.2 are much better than the results of using the image watermark. Hence, the text watermarks are more robust than image watermarks to the editing and deletion attacks.

#### 4.4 Comparison with existing methods

We compare our proposed method with two recent watermarking methods in [20] and [17]. In our comparison, we focus on editing and deletion attacks. The method of Yang et al. [17] is based on vector maps, so our comparative experiments only use the vector graphs of the legend library, which include 327 vector point legends, 256 vector line legends, and 47 vector surface legends. To have a quantitative comparison, we adopt the correlation coefficient metric used in [17] for comparing the extracted watermarks and the embedded counterparts. A threshold is usually used to decide if a watermark is successfully recovered [17]. That is, if a correlation coefficient is above a threshold, the recovered watermark is considered intact (or verifiable); otherwise, it is not verifiable. Our results are reported in the following two tables and the cases that fail below the threshold are highlighted in blue.

Table 10 lists the correlation coefficients between the original watermarks and the extracted watermarks after legend editing attacks. The last column lists the average correlation coefficients for the case. As the amount of edits increases, the average correlation coefficient decreases for both compared methods. On average, the method by Yang et al. [17] exhibits a slightly better performance than the method by Zhou et al. [20]. The watermark generated by the proposed method successfully extracts the watermarks under different amounts of editing attacks and the correlation coefficients are at the maximum of one throughout. The method by Yang et al. [17] successfully extracts the watermarks from point legends in all cases. However, cases below the threshold appear when the amount of edits is at 90% for both line and area legends. The cause of the failure of extracting watermark in line legends is that the distribution of legend data (i.e., coordinates) is in a small range, which makes it difficult to have enough bit space to embed all watermark bits. A similar failure is observed in the results by the method of Zhou et al. [20]. There exists one case that is below the threshold and several cases that are barely above the threshold at 90%. The method suffers from dependencies among multiple extracted watermarks, which leads to incomplete extracted watermarks.



Table 11 presents the correlation coefficients between the original watermarks and the extracted watermarks after legend deletion attacks. The last column lists the average correlation coefficients for the case. We observe a very similar pattern to the one in the previous table. In addition to the decreasing correlation coefficients of the two compared methods as

**Table 9** The extracted image watermark after deletion attacks

Deletion		Point legend	Line legend	Area legend
Average deletion	30%	CUG MapGIS 25w		
	45%	CUG MapGIS 25w		
	60%	CUG MapGIS 25w		
	75%	CUG MapGIS 25w		
	90%			
Random deletion	30%	CUG MapGIS 25w		
	45%	CUG MapGIS 25w		
	60%	CUG MapGIS 25w		
	75%	CUG MapGIS 25w		
	90%			
	30%	CUG MapGIS 25w		

**Table 9** (continued)

Deletion		Point legend	Line legend	Area legend
Mass deletion	45%	CUG MapGIS 25w		
	60%			CUG MapGIS 25w
	75%			
	90%			

the amount of deletion increases, we observe cases that fall below the threshold at 90%. In our watermark embedding method, we take into consideration of extreme situations including editing all bits with watermark information or editing the primitives of the legend, which causes the loss of the watermark bits in the map legends.

**Table 10** The correlation coefficient of the extracted watermarks after editing attacks

Method	Average editing	Random editing			Mass editing						
		Point	Line	Area	Point	Line	Area	Point	Line	Area	Ave.
30%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.96	0.95	0.9	0.94	0.94	0.87	0.97	0.94	0.88	0.93
	Yang [17]	1	1	1	1	1	1	1	1	1	1
45%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.91	0.88	0.84	0.93	0.86	0.84	0.93	0.86	0.82	0.87
	Yang [17]	1	1	1	1	1	1	1	1	1	1
60%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.87	0.82	0.76	0.87	0.83	0.77	0.87	0.83	0.74	0.82
	Yang [17]	1	0.92	0.98	1	0.87	0.92	1	0.86	0.96	0.95
75%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.84	0.76	0.67	0.82	0.71	0.68	0.82	0.71	0.56	0.73
	Yang [17]	1	0.74	0.82	1	0.66	0.79	1	0.75	0.76	0.84
90%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.79	0.65	0.56	0.77	0.62	0.59	0.77	0.55	0.45	0.63
	Yang [17]	1	0.47	0.51	1	0.47	0.53	1	0.48	0.46	0.66

**Table 11** The correlation coefficient of the extracted watermarks after deletion attacks

	Method	Average deletion			Random deletion			Mass deletion			Ave.
		Point	Line	Area	Point	Line	Area	Point	Line	Area	
30%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.96	0.95	0.9	0.95	0.93	0.88	0.97	0.92	0.84	0.92
	Yang [17]	1	1	1	1	1	1	1	1	1	1
45%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.91	0.88	0.84	0.92	0.85	0.82	0.93	0.87	0.79	0.87
	Yang [17]	1	1	1	1	1	1	1	1	1	1
60%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.87	0.82	0.76	0.86	0.82	0.76	0.87	0.81	0.69	0.81
	Yang [17]	1	0.92	0.98	1	0.89	0.91	1	0.94	0.96	0.96
75%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.84	0.76	0.67	0.82	0.75	0.7	0.82	0.71	0.56	0.74
	Yang [17]	1	0.74	0.82	1	0.68	0.77	1	0.75	0.76	0.84
90%	Proposed	1	1	1	1	1	1	1	1	1	1
	Zhou [20]	0.79	0.65	0.56	0.78	0.65	0.58	0.77	0.58	0.45	0.65
	Yang [17]	1	0.47	0.51	1	0.42	0.59	1	0.48	0.46	0.66

## 5 Conclusion

Map legends are a kind of special hybrid data. This paper presents a watermark bit recovery method to improve the robustness of the embedded watermark that embeds multiple copies of the watermark to the host data. Our method uses text watermarks instead of the conventional image watermarks to reduce the number of bits altered in the embedding process, which makes it possible to embed more watermark copies in a map legend library. The watermark bits embedding position is based on Just Noticeable Distortion to make the embedded watermarks imperceptible.

We evaluate our method using the legend library of MapGIS K9 for robustness to various attacks as well as visibility. Our experimental results demonstrate that the proposed is robust to the common attacks to map legends including moving, adding, editing, and deletion attacks. In contrast to image watermarks, the text watermarks embedded using our method take much less space and, hence, multiple copies can be embedded in the map legends. In the comparison study with the existing methods, our method exhibits superior performance in terms of correlation coefficients.

**Funding** This work was supported by the National Natural Science Foundation of China, grant number 41871305, and the National Key R & D Program of China, grant number 2017YFC0602204.

## Declarations

**Competing interests** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Abubahia A, Cocea M (2017) Advancements in gis map copyright protection schemes - a critical review. *Multimed Tools Appl* 76:12205–12231
2. Cox IJ, Miller ML, Bloom JA, Fridrich J, Kalker T, Digital watermarking and steganography (2nd edn.). Morgan Kaufmann Burlington pp 1–13 (2008) Introduction. In: Cox IJ, Miller ML, Bloom JA, Fridrich J, Kalker T (eds)
3. Dyson FJ, Falk H (1992) Period of a discrete cat mapping. *American Math Month* 99(7):603–614
4. Gomathisankaran M, Yuan X, Kamongi P (2013) Ensure privacy and security in the process of medical image analysis. In: 2013 IEEE international conference on granular computing (GrC), pp 120–125
5. Haitao Z, Xuefang L, Xiaorong G (2013) Research on the methods of quality inspection for published charts based on entity rules and cartographic specifications. *journal=Int Arch Photogrammetry Remote Sensing Spatial Inf Sci*, XL-2/W1:65–69
6. Hamouda E, Yuan X, Ouda O, Hamza T, Chen L (2014) Binary biometric template generation towards security and class separability. In: Fifth international conference on computing, communications and networking technologies, pp 1–6
7. Horng S-J, Rosiyadi D, Fan P, Wang X, Khan K (2014) An adaptive watermarking scheme for e-government document images. *Multimed Tools Appl* 72:10
8. Jinxia W, Yang F, Wu C (2013) Review of digital watermarking for 2d-vector map. In: 2013 IEEE international conference on green computing and communications and IEEE internet of things and IEEE Cyber, physical and social computing, pp 2098–2101
9. Katzenbeisser S, Petitcolas FA (2000) *Information Hiding Techniques for Steganography and Digital Watermarking*, 1st edn. Artech House, Inc., USA
10. Kim J, Won S, Zeng W, Park S (2011) Copyright protection of vector map using digital watermarking in the spatial domain. In: The 7th international conference on digital content, multimedia technology and its applications, pp 154–159
11. Li S-S, Zhou W, Li A-B (2012) Image watermark similarity calculation of gis vector data. *Procedia Eng* 29:1331–1337
12. Lv Y-l, Wang R-f, Zhao S-j, Wang X, Yang C-h (2010) Design and development of scales and slope scales meeting national specifications based on arcgis platform. In: 2010 second IITA international conference on geoscience and remote sensing, vol 1, pp 560–563
13. Rosiyadi D, Horng S-J, Lestriandoko NH (2015) A resistant digital image watermarking scheme based on masking model. In: 2015 international carnahan conference on security technology (ICCST), pp 1–4
14. Wang L, Fang F, Yuan X, Luo Z, Liu Y, Bo W, Zhao Y (2017) Urban function zoning using geotagged photos and openstreetmap. In: 2017 IEEE international geoscience and remote sensing symposium (IGARSS), pp 815–818
15. Wen Q, Sun T-F, Wang S-X (2003) Concept and application of zero-watermark. *Acta Electron Sin* 31:214–216
16. Wu M, Zhu A, Zheng P, Cui L, Zhang X (2017) An improved map-symbol model to facilitate sharing of heterogeneous qualitative map symbols. *Cartogr Geogr Inf Sci* 44(1):62–75
17. Yang C, Zhu C, Wang Y, Rui T, Zhu J, Kaimeng D (2020) A robust watermarking algorithm for vector geographic data based on qim and matching detection. *Multimed Tools Appl* 79:11
18. Yuan X, Gomathisankaran M (2016) *Secure medical image processing for mobile devices 341 using cloud services*, chapter 6. SPIE PRESS, pp 155–174
19. Zhen Z (2012) Automated plotting technology research for the point symbols in the multi-core environment. In: 2012 20th international conference on geoinformatics, pp 1–6
20. Zhou L, Chen ZL, Wu L (2017) A zero-watermarking algorithm of map legend library based on item characteristics. *Sci Surveying Mapp* 42(3):137–142
21. Zhou G, Zeng P, Yuan X, Chen S, Choo K-KR (2017) An efficient code-based threshold ring signature scheme with a leader-participant model. *Sec Commun Netwo*:2017

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.