http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 13 (5): 869-873, 2014
ISSN 1812-5638 / DOI: 10.3923/itj.2014.869.873

A Blind Watermarking for 2D-Vector Engineering Graphics

Handan Hou, Jian Li, Jingjia Qi and Junfeng Guo
School of Computer Science, Harbin Finance University, Harbin, Xiangfang, Heilongjiang, China

Abstract: A blind method for watermarking of 2D-vector engineering graphics based on distance of each two adjacent vertices is proposed in this study. The watermarking algorithm has high capacity to embed enough information for copyright protection. Watermarks are embedded within the tolerance of the graphics, leading to invisible distortions to the coordinates of vertices. The method is resilient to addition/deletion entities, cropping, translation and rotation.

Key words: 2D-vector engineering graphics, blind, watermarking

INTRODUCTION

Engineering graphics, more efficient with high precision, little data and less redundant data, are playing an important role in manufacturing, architecture and electrical fields. Thanks to the hard work of the engineers, these graphics are of great worth. They can be, however, easily duplicated and distributed in a network environment. As a technique of copyright protection, watermarking receives a lot of attention recently.

Despite the importance of vector graphics, most algorithms focus on applications of watermarking in raster images such as images, audios, videos and texts. Methods of vector watermarking are mainly applied in 3D vector graphics (Zafeiriou *et al*., 2005; Bors, 2006; Cho *et al*., 2007) while little work has been done in watermarking of 2D vector graphics data especially for engineering graphics. Voigt and Busch (2002), Ohbuchi *et al*. (2003), Voigt *et al*. (2004) and Wang *et al*. (2007) proposed watermarking algorithms for vector maps in which distances exhibit high correlation. Most of these methods have limited robustness regarding translation, rotation or scaling since such attacks are not usually encountered for GIS data. Thus, these algorithms might not be applicable to engineering graphics where translation, rotation and scaling are common manipulation. Sonnet *et al*. (2003) introduced algorithms that change line attributes, introduce new vertices in certain patterns and replace existing stroke segments by new lines in a stylistic way. The proposed methods seem to be robust to format conversion, translation, rotation, scaling and cropping. However, the watermarks can be removed easily by attacks designed specifically for each method. Solachidis and Pitas (2004) proposed a blind method for watermarking of vector graphics through polygonal line modification. Properties of the Fourier descriptors ensure that the algorithm can withstand rotation, translation, scaling, reflection and smoothing. But it is fragile to local modification and is not applicable to graphics in which polygonal lines contain few vertices. Similar to the work of Solachidis and Pitas (2004) and, Zhang *et al*. (2005) embedded watermark in the complex wavelet domain of relative coordinate lines. It is robust as Solachidis' method and is additionally robust to local modification. However, it cannot be applied to graphics in which polygonal lines containing few vertices.

This study presents a method for robust and blind watermarking of two-dimensional engineering graph by embedding watermarks in the distance of each two adjacent vertices.

ALGORITHM DESCRIPTION

Basic idea of the algorithm: In this study, the raw coordinates of vertices in the original graphics are used as the cover data for hiding watermarks. Given v_i and v_{i+1} are two adjacent vertices with coordinates of (x_i, y_i) and (x_{i+1}, y_{i+1}). L_i is the distance between v_i and v_{i+1} which is defined in Eq. 1:

$$L_i = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \quad (1)$$

Embed the ith bit of watermarks by modifying the s least significant decimal digits of L_i. Suppose that the new distance carrying the ith bit of watermarks is denoted as L'_i. The watermarked elements $(x'_i, 0y'_i)$ and (x'_{i+1}, y'_{i+1}) can be calculated via Eq. 2:

Corresponding Author: Handan Hou, School of Computer Science, Harbin Finance University, Harbin, Xiangfang, Heilongjiang, China

$$\begin{cases} x_i' = x_i \\ y_i' = y_i \\ x_{i+1}' = x_i' - \frac{L_i'}{L_i}(x_i - x_{i+1}) \\ y_{i+1}' = y_i' - \frac{L_i'}{L_i}(y_i - y_{i+1}) \end{cases} \quad (2)$$

Watermarks generation: The data to be embedded is an m-dimensional bit vector $a = (a_1, a_2,\ldots, a_m)$ in which each bit takes values {0,1}. Each bit a_i is spread spatially over the graphics by duplicating each symbol by chip rate c, producing a watermark symbol vector $b = (b_1, b_2,\ldots, b_{mc})$, $b_i \in \{0,1\}$ of length m.c. Repeatedly embedding the same bit c times increases resiliency of the watermark against cropping, vertex insertion and deletion.

Watermarking embedding:

- **Extraction of the vertices:** The vertices are extracted from entities of the graphics which consist of points, start points and end points of lines, positions of texts, center points of circles and ellipses and etc. The vertices are represented as $v = (v_1, v_2,\ldots, v_n)$ and put in order according to the order of corresponding entities. N presents the number of the vertices. Each of the vertices is represented as a pair of coordinates (x_k, y_k)
- **Modification of s:** Embedding is achieved by modifying the s least significant decimal digits of L_i, so we should fix on the value of s. Normally, the coordinates of a vector graphics are floating-point numbers with fixed precision. We choose s as the decimal digit position of the coordinate value where changes are significant but not violating the tolerance requirements

For example, the precision of a vector graphics is 0.00000000 and then s will be set as 9. It means that the 9th decimal digit after the decimal point of L_i will be changed when embedding.

- **Embedding watermark into L_i:** To embed watermark b_i, we firstly calculate the distance L_i between v_i and v_{i+1} according Eq. 1. If $b_i = 0$, change the s least significant decimal digits of L_i to 2. Otherwise, if $b_i = 1$, the s least significant decimal digits of L_i is changed to 7. Thus, we get the new distance L'_i

Take $L_i = 1.25896548568$ for example. If $b_i = 0$, $L'_i = 1.25896548268$. Otherwise, if $b_i = 1$, $L'_i = 1.25896548768$.

- **Changing the coordinates of v_i and v_{i+1}:** The watermarked coordinates (x'_i, y'_i) and (x'_{i+1}, y'_{i+1}) can then be calculated via Eq. 2

Watermarking extraction: The step 1 and 2 are similar to watermarking embedding. Then we get the vertices sequence in order represented as $v' = \{v'_1, v'_2,\ldots, v'_N\}$. The value of s is also set according to the precision of the watermarked graphics. Then, we calculate the distance L'_i between v'_i and v'_{i+1}. Denote the s least significant decimal digits of L'_i as $D^s_{L'_i}$. If $D^s_{L'_i} < 5$, set $b'_i = 0$. Otherwise, if $D^s_{L'_i} \geq 5$, set $b'_i = 1$. In the same manner, we can get the spread watermark symbol vector $b' = (b'_1, b'_2,\ldots, b'_{mc})$, $b'_i = 1$. Via statistical method, the m-dimensional watermarks $a' = (a'_1, a'_2,\ldots, a'_m)$ can be got from vector b'.

EXPERIMENTS AND RESULTS

Graphics of DWG format are used as the original graphics to test the performance of the proposed algorithm which is represented as G in this section. The graphics contain 12307 entities and 18275 vertices. The length of the watermarks to be embedded is 84(bit).

This section will show experiments, results and analysis on the capacity, invisibility and robustness of the proposed algorithm.

Capacity: From water marking embedding, we can know that the capacity is contributed by vertices of entities in the graphics. Suppose the number of vertices is N_V, the space provided should be

$$N_V\text{-}1 \quad (3)$$

Thus, the capacity of a graphics is determined by Eq. 3. More vertices will result in higher capacity. The experimental result is shown in Table 1. The capacity is very high with the data of 1.48 bits per entity and 1.00 bit per vertex.

Invisibility: Five engineering graphics, with the precision of 0.00000000, are used to test the performance of invisibility. Figure 1 shows details of an original graphic and watermarked graphic. Table 2 exhibits the results of experiments on invisibility, in which the Mean Error (ME) is introduced as the distortion induced by the algorithm. Table 2 shows that MEs of the 5 graphics are all less than the precision which implies that the distortion induced by the algorithm is low and acceptable.

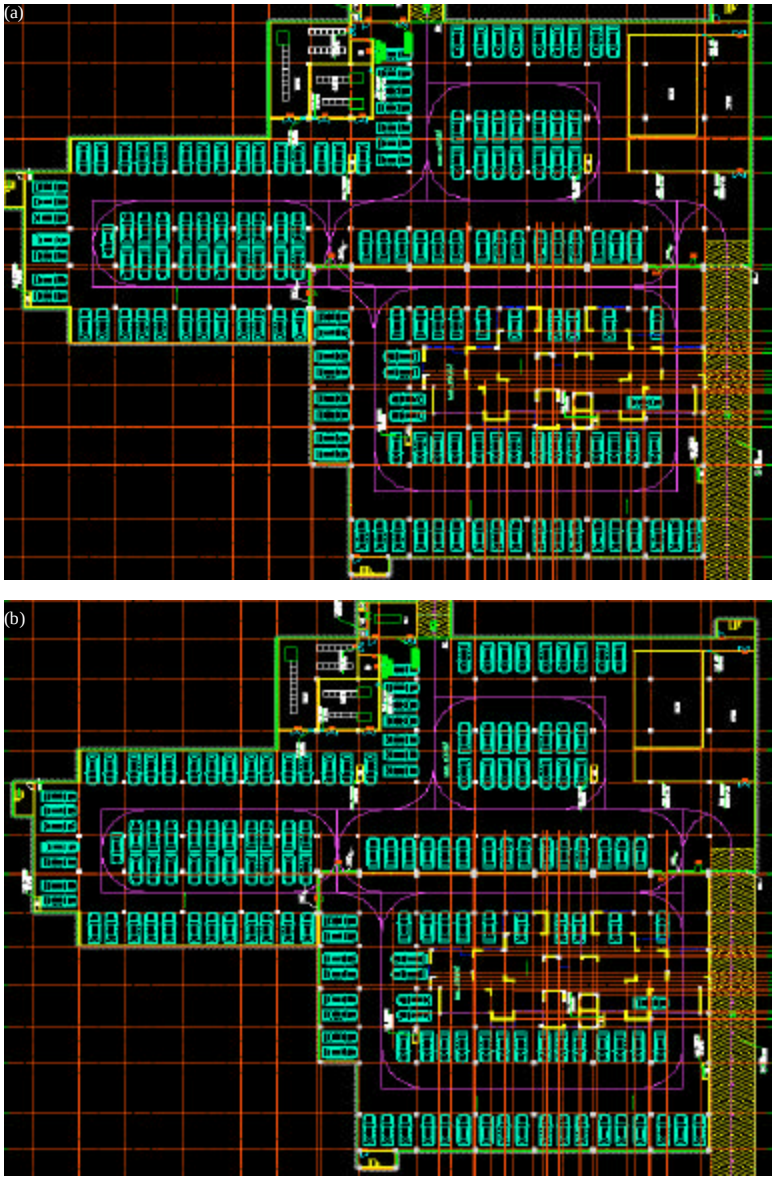

Fig. 1(a-b): (a) Original graphics and (b) Watermarked graphics

Table 1: Capacity of a graphics

Entities	Vertices	Watermarks (bit)	Bits/entity	Bits/vertex
12307	18275	18274	1.48	1.00

Table 2: Invisibility of graphics

Index	File size (MB)	Entities	Vertices	ME
1	1.47	12307	18275	1.8663E-09
2	1.73	44405	82080	1.7206E-09
3	1.92	14531	21967	1.8318E-09
4	2.00	26721	42628	1.8402E-09
5	3.29	20169	28690	1.8290E-09

Robustness: Various attacks have to be faced for watermark of engineering graphics. In this section, experiments will be carried out to evaluate the robustness of the algorithm against addition/deletion entities, cropping, translation and rotation. Graphics G' is the watermarked graphics of G and used to test the performance of robustness.

Addition/deletion entities: After addition or deleting some entities of the graphics, watermarks embedded will be still decodable since each bit of the watermarks was embedded repeatedly. Here take deletion for example. Via deleting some kinds of entities, a number of vertices are deleted randomly. Table 3 shows the results which indicate that the algorithm is resilient against deletion.

Table 3: Tests of deletion entities

Entities deleted	No. of deleted entities	Deleted vertices	Erroneous watermarks
Walls	231	413	0
Red entities	1154	3368	0
Blue entities	2108	4216	0
Blue and red entities	3262	7584	0
Blue, red and green entities	6352	10640	0

Table 4: Tests of cropping

Ratio of cropping	Removed entities	Removed vertices	Erroneous watermarks
1/14	803	1131	0
5/14	4439	6705	0
9/14	6191	8557	0
27/28	10886	16727	0

Table 5: Tests of translation and rotation

(Δx, Δy)	Erroneous watermarks	Δθ	Erroneous watermarks
(10, 20)	0	30°	0
(2000, 1000)	0	90°	0
(-5000, -10000)	0	-60°	0
(10000, 20000)	0	-120°	0

Cropping: As shown in Table 4, we crop graphics G' with various ratios and use the remaining as the test graphics. From the results it is easily seen that the algorithm is robust against cropping.

Translation: Take two adjacent vertices $v'_i(x'_i, y'_i)$ and $v'_{i+1}(x'_{i+1}, y'_{i+1})$ for example. After translating graphics G' by vector (Δx, Δy), coordinates of the two vertices will be $(x'_i+\Delta x, y'_i+\Delta y)$ and $(x'_{i+1}+\Delta x, y'_{i+1}+\Delta y)$. L'^{T}_i is defined as the new distance between v'_i and v'_{i+1}:

$$L'^{T}_i = \sqrt{\left[\left(x'_i+\Delta x\right)-\left(x'_{i+1}+\Delta x\right)\right]^2+\left[\left(y'_i+\Delta y\right)-\left(y'_{i+1}+\Delta y\right)\right]^2} = \sqrt{\left(x'_i-x'_{i+1}\right)^2+\left(y'_i-y'_{i+1}\right)^2} = L'_i \quad (4)$$

Since translation does not affect the distance between two vertices, the algorithm is resilient against translation. We have done some experiments with results in Table 5 to prove the robustness of translation.

Rotation: In the polar coordinate system, vertices v'_i and v'_{i+1} can be described as (ρ'_i, θ'_i) and $(\rho'_{i+1}, \theta'_{i+1})$. Then the distance between v'_i and v'_{i+1} can be calculated via Eq. 5:

$$L'_i = \sqrt{\left(\rho'_i\right)^2+\left(\rho'_{i+1}\right)^2-2\rho'_i\rho'_{i+1}\cos\left(\theta'_i-\theta'_{i+1}\right)} \quad (5)$$

Rotate graphics G' by Δθ anticlockwise. Then coordinates of v'_i and v'_{i+1} will be $(\rho'_i, \theta'_i+\Delta\theta)$ and $(\rho'_{i+1}, \theta'_{i+1}+\Delta\theta)$. The new distance L'^{R}_i can be calculated by Eq. 6:

$$L'^{R}_i = \sqrt{\left(\rho'_i\right)^2+\left(\rho'_{i+1}\right)^2-2\rho'_i\rho'_{i+1}\cos\left[\left(\theta'_i+\Delta\theta\right)-\left(\theta'_{i+1}+\Delta\theta\right)\right]} = \sqrt{\left(\rho'_i\right)^2+\left(\rho'_{i+1}\right)^2-2\rho'_i\rho'_{i+1}\cos\left(\theta'_i-\theta'_{i+1}\right)} = L'_i \quad (6)$$

From Eq. 6 we can see that rotation does not change the distance between two vertices, the algorithm is robust against rotation. Table 5 shows some experiments and results on robustness of rotation to support our analysis.

CONCLUSION

A blind watermarking method for 2D-vector engineering graphics is proposed in this study. The algorithm is based on distance of each two adjacent vertices and embeds watermarks repeatedly. As shown in experiments and results, the watermarking algorithm has high capacity, is invisible and robust to addition/deletion entities, cropping, translation and rotation while still robust to scaling. which will be the focus of future work.

ACKNOWLEDGMENT

This study is supported by GZ11A302 of Heilongjiang Province, China.

REFERENCES

Bors, A.G., 2006. Watermarking mesh-based representations of 3-D objects using local moments. IEEE Trans. Image Process., 15: 687-701.

Cho, J.W., R. Prost and H.Y. Jung, 2007. An oblivious watermarking for 3-D polygonal meshes using distribution of vertex norms. IEEE Trans. Signal Process., 55: 142-155.

Ohbuchi, R., H. Ueda and S. Endoh, 2003. Watermarking 2D vector maps in the mesh-spectral domain. Proceedings of the Shape Modeling International, May 12-15, 2003, Los Alamitos, CA., USA., pp: 216-225.

Solachidis, V. and I. Pitas, 2004. Watermarking polygonal lines using Fourier descriptors. Comput. Graph. Appl., 24: 44-51.

Sonnet, H., T. Isenberg, J. Dittmann and T. Strothotte, 2003. Illustration watermarks for vector graphics. Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, October 8-10, 2003, Magdeburg, Germany, pp: 73-82.

Voigt, M. and C. Busch, 2002. Watermarking 2D-vector data for geographical information systems. Proceedings of the Security and Watermarking of Multimedia Contents IV, April 29, 2002, San Jose, CA.

Voigt, M., B. Yang and C. Busch, 2004. Reversible watermarking of 2D-vector data. Proceedings of the Workshop on Multimedia and Security, September 20-21, 2004, Magdeburg, Germany, pp: 160-165.

Wang, X., C. Shao, X. Xu and X. Niu, 2007. Reversible data-hiding scheme for 2-D vector maps based on difference expansion. IEEE Trans. Inform. Forensics Security, 2: 311-320.

Zafeiriou, S., A. Tefas and I. Pitas, 2005. Blind robust watermarking schemes for copyright protection of 3D mesh objects. IEEE Trans. Viusal. Comput. Graph., 11: 596-607.

Zhang, Q., H. Xiang and X.X. Meng, 2005. Watermarking vector graphics based on complex wavelet transform. J. Image Graph., 10: 494-498.