Fingerprinting Curves

Hongmei Gou and Min Wu

University of Maryland, College Park, U.S.A.*

Abstract. This paper presents a new method for robust data hiding in curves and highlights potential applications including digital fingerprinting of map document for trace and track purposes. We parameterize a curve using the B-spline model and add a spread spectrum sequence to the coordinates of the B-spline control points. In order to achieve robust fingerprint detection, we propose an iterative alignment-minimization algorithm to perform curve registration and deal with the non-uniqueness of B-spline control points. We show through experiments the robustness of our method against various attacks such as collusion, geometric transformations and perturbation, printing-and-scanning, and some of their combinations. We demonstrate the feasibility of our method for fingerprinting topographic maps as well as writings and drawings. The extension from hiding data in 2D topographic maps to 3D elevation data sets is also discussed.

1 Introduction

Map represents geospatial information ubiquitous in government, military, intelligence, and commercial operations. The traditional way to protecting map from unauthorized copying and distribution is to place deliberate errors in the map, such as spelling "Nelson Road" as "Nelsen Road", bending a road in a wrong way, and/or placing a non-existing pond. If an unauthorized user has a map containing basically the same set of errors, this is a strong piece of evidence on piracy that can be presented in court. However, the traditional protection methods alter the geospatial meanings conveyed by a map, which can cause serious problems in critical government, military, intelligence, and commercial operations where highly precise geospatial information is needed. Further, in the situations where the distinct errors serve as fingerprints to trace individual copies, such intentional errors can be rather easily identified and removed by computer algorithms after multiple copies of a map are brought to the digital domain. All these limitations of the traditional methods prompt for a modern way to map protection that can be more effective and less intrusive.

Curve is one of the major components appearing in maps as well as other documents such as drawings and signatures. A huge amount of curve-based documents are being brought to the digital domain owing to the popularity of scanning devices and pen-based devices (such as TabletPC). Digital maps

* The authors can be contacted via email at {hmgou, minwu}@eng.umd.edu.

I.J. Cox et al. (Eds.): IWDW 2004, LNCS 3304, pp. 13–28, 2005.

and drawings are also generated directly by various computer programs such as map-making software and CAD systems. Having the capability of hiding digital watermarks or other secondary data in curves can facilitate digital rights management of important documents in government, military, intelligence, and commercial operations. For example, trace-and-track capabilities can be provided through invisibly embedding a unique ID, referred to as a *digital fingerprint*, to each copy of a document before distributing to users [1]. In this paper, we present a new, robust data hiding technique for curves and investigate its feasibility for fingerprinting maps.

As a forensic mechanism to deter information leak and to trace traitors, digital fingerprint must be difficult to remove. For maps and other visual documents, the fingerprint has to be embedded in a robust way against common processing and malicious attacks. Some examples include collusion, where several users combine information from different copies to generate a new copy in which the original fingerprints are removed or attenuated [1]; various geometric transformations such as rotation, scaling, and translation (RST); and D/A-A/D conversions such as printing-and-scanning. On the other hand, the fingerprint must be embedded in a visually non-intrusive way without changing the geographical and/or visual meanings conveyed by the document. Such changes may have serious consequences in critical military and commercial operations, for example, when inaccurate data are given to troops or fed into navigation systems.

There is a very limited amount of existing work on watermarking maps [2], and few exploits curve features or addresses fingerprinting issues. A text-based geometric normalization method was proposed in [3], whereby text labels are used to normalize the orientation and scale of the map image and conventional robust watermarking algorithms for grayscale images are then applied. As maps can be represented as a set of vectors, two related works on watermarking vector graphics perturb vertices through Fourier descriptors of polygonal lines [4] or spectral analysis of mesh models [5] to embed copyright marks. The embedding in [4] introduces visible distortions, as shown in their experimental results. The watermarking approach in [5] has high complexity resulting from the mesh spectral analysis. Besides, it cannot be easily applied to maps beyond urban areas, where curves serve as an essential component in mapping a vast amount of land and underwater terrains. Since curve-based documents can also be represented as binary bitmap images (known as the raster representation), we expand the literature survey to data embedding works for general binary images. The data hiding algorithm in [6] enforces the ratio of black versus white pixels in a block to be larger or smaller than 1, and the "flippable" pixels are defined and used to enforce specific block-based relationship to embed data in [7][8]. The fragility of the embedding and the reliance on precise sampling of pixels for correct decoding pose challenges in surviving geometric transformations, printing-and-scanning, and malicious removal in fingerprinting applications.

There are several watermarking algorithms for graphic data employing parametric features, such as representing 3D surfaces by the non-uniform rational B-spline (NURBS) model and changing the NURBS knots or control points to

embed data [9][10]. While these works provide enlightening analogy for watermarking the 2D counterpart (i.e. curves) in the B-spline feature domain, most of the existing explorations and results are for fragile embedding in 3D surfaces and have limited robustness. There has been few discussion on robust watermarking of curves, and to our knowledge, no existing work has demonstrated the robustness against curve format conversion and D/A-A/D conversion.

In this paper, we propose a robust curve watermarking method and apply it to fingerprinting maps without interfering with the geospatial meanings conveyed by the map. We select B-spline control points of curves as the feature domain and add mutually orthogonal, noise-like sequences as digital fingerprints to the coordinates of the control points. A proper set of B-spline control points forms a compact collection of salient features representing the shape of the curve, which is analogous to the perceptually significant spectral components in the continuous-tone images [11]. The shape of curves is also invariant to such challenging attacks as printing-and-scanning and the vector/raster-raster/vector conversion. The additive spread spectrum embedding and the corresponding correlation based detection generally provide a good tradeoff between imperceptibility and robustness [11]. To determine which fingerprint sequence(s) are present in a test curve, registration with the original unmarked curve is an indispensable preprocessing step. B-splines have invariance to affine transformation in that the affine transformation of a curve is equivalent to the affine transformation of its control points. This affine invariance property of B-splines can facilitate automatic curve registration. Meanwhile, as a curve can be approximated by different sets of B-spline control points, we propose an iterative alignment-minimization (IAM) algorithm to simultaneously align the curves and identify the corresponding control points. Through the B-spline based data hiding plus the IAM algorithm for robust fingerprint detection, our curve watermarking technique can sustain a number of challenging attacks such as collusion, geometric transformations, vector/raster-raster/vector conversions, and printing-and-scanning, and is therefore capable of building collusion-resistant fingerprinting for maps and other curve-based documents.

The paper is organized as follows. Section 2 discusses the feature domain in which data hiding is performed and presents the basic embedding and detection algorithms with experimental results on marking simple curves. Section 3 details the proposed iterative alignment-minimization algorithm for the fingerprint detection and analyzes its robustness. Experimental results on fingerprinting topographic maps are presented in Section 4 to demonstrate the robustness of our method against a number of processing and attacks. Section 5 extends our curve watermarking method to protecting 3D geospatial data set. Finally conclusions are drawn in Section 6.

2 Basic Embedding and Detection

We first present the basic embedding and detection scheme on curves. We employ the coordinates of B-spline control points as the embedding feature domain,

and adopt spread spectrum embedding [11] and correlation-based detection for watermarking curves.

2.1 Feature Extraction

A number of approaches have been proposed for curve modelling, including using the chain codes, the Fourier descriptors, the autoregressive models, and the B-splines [12]. Among them, the B-splines are particularly attractive and have been extensively used in computer-aided design and computer graphics. This is mainly because the B-spline model provides a bounded and continuous approximation of a curve with excellent local shape control and is invariant to affine transformations [13]. These advantages also lead to our choosing B-splines as the feature domain for data embedding in curves.

B-splines are piecewise polynomial functions that provide local approximations of curves using a small number of parameters known as the *control points* [12]. Let $\{\mathbf{p}(t)\}$ denote a curve, where $\mathbf{p}(t) = (p_x(t), p_y(t))$ and t is a continuous time variable. Its B-spline approximation $\{\mathbf{p}^{[B]}(t)\}$ can be written as

$$\mathbf{p}^{[B]}(t) = \sum_{i=0}^{n} \mathbf{c}_i B_{i,k}(t), \tag{1}$$

where $\mathbf{c}_i = (c_{x_i}, c_{y_i})$ is the i^{th} control point ($i = 0, 1, \ldots, n$), t ranges from 0 to $n-1$, and $B_{i,k}(t)$, the weight of the i^{th} control point for the point $\mathbf{p}^{[B]}(t)$, is a corresponding k^{th} order B-spline blending function recursively defined as

$$\begin{aligned} B_{i,1}(t) &= \begin{cases} 1 & t_i \le t < t_{i+1} \\ 0 & \text{otherwise,} \end{cases} \\ B_{i,k}(t) &= \frac{(t - t_i) B_{i,k-1}(t)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - t) B_{i+1,k-1}(t)}{t_{i+k} - t_{i+1}} \qquad k = 2, 3, \ldots \end{aligned} \tag{2}$$

where $\{t_i\}$ are parameters known as *knots* and represent locations where the B-spline functions are tied together [12]. The placement of knots controls the form of B-spline functions and in turn the control points.

As a compact representation, the number of B-spline control points necessary to represent the curve at a desired precision can be much smaller than the number of sample points from the curve typically obtained through uniform sampling. Thus, given a set of samples on the curve, finding a smaller set of control points for its B-spline approximation that minimizes the approximation error to the original curve can be formulated as a least-squares problem. Coordinates of the $m+1$ samples on the curve can be represented as an $(m+1) \times 2$ matrix of $\mathbf{P} \stackrel{\triangle}{=} (\mathbf{p}_x, \mathbf{p}_y)$. The time variables of the B-spline blending functions corresponding to these $m+1$ samples are $t = s_0, s_1, s_2, \ldots, s_m$, where $s_0 < s_1 < s_2 < \ldots < s_m$. Further, let $\mathbf{C} \stackrel{\triangle}{=} (\mathbf{c}_x, \mathbf{c}_y)$ represent the coordinates of $n+1$ control points. Then we can write the least-squares problem with its solution as

$$\mathbf{BC} \approx \mathbf{P} \quad \Longrightarrow \quad \mathbf{C} = \left(\mathbf{B^T B}\right)^{-1} \mathbf{B^T P} = \mathbf{B}^{\dagger} \mathbf{P}, \tag{3}$$

where $\{\mathbf{B}\}_{ji}$ is the value of the k^{th}-order B-spline blending function $B_{i,k}(t)$ in (2) evaluated at $t = s_j$ and † denotes the pseudo inverse of a matrix. Due to the natural decoupling of the x and y coordinates in the B-spline representation, we can solve the problem separately along each of the two coordinates as

$$\begin{cases} \mathbf{B}\mathbf{c}_x \approx \mathbf{p}_x \\ \mathbf{B}\mathbf{c}_y \approx \mathbf{p}_y \end{cases} \implies \begin{cases} \mathbf{c}_x = \mathbf{B}^\dagger \mathbf{p}_x \\ \mathbf{c}_y = \mathbf{B}^\dagger \mathbf{p}_y \end{cases}. \tag{4}$$

2.2 Data Embedding and Detection in Control Points

The control points of a curve are analogous to the perceptually significant spectral components of a continuous-tone image [11] in that they form a compact set of salient features for curve. In such a feature domain, we apply spread spectrum embedding and correlation based detection.

The spread spectrum embedding generally offers a good tradeoff between imperceptibility and robustness, especially when the original host signal is available to the detector as in most of the fingerprinting applications [1]. In the embedding, we use mutually orthogonal, noise-like sequences as digital fingerprints to represent different users/IDs for trace and track purposes. As each of the $n+1$ control points has two coordinate values x and y, the overall length of the fingerprint sequence is $2(n+1)$. To apply spread spectrum embedding on a curve, we add a scaled version of the fingerprint sequence $(\mathbf{w}_x, \mathbf{w}_y)$ to the coordinates of a set of control points obtained from the previous subsection. This results in a set of watermarked control points $\left(\mathbf{c}'_x, \mathbf{c}'_y\right)$ with

$$\begin{cases} \mathbf{c}'_x = \mathbf{c}_x + \alpha \mathbf{w}_x \\ \mathbf{c}'_y = \mathbf{c}_y + \alpha \mathbf{w}_y \end{cases}, \tag{5}$$

where α is a scaling factor adjusting the fingerprint strength. A watermarked curve can then be constructed by the B-spline synthesis equation (1) using these watermarked control points.

To determine which fingerprint sequence(s) are present in a test curve, we first need to perform registration using the original unmarked curve that is commonly available to a detector in fingerprinting applications. After registration, control points $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$ are extracted from the test curve. The accurate registration and correct extraction of control points are crucial to the accurate detection of fingerprints, which will be detailed in Section 3. Assuming we have the set of sample points given by $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y) = (\mathbf{B}(\mathbf{c}_x + \alpha \mathbf{w}_x), \mathbf{B}(\mathbf{c}_y + \alpha \mathbf{w}_y))$, we can extract the test control points $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$ from $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ using (4). After getting $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$, we compute the difference between the coordinates of the test and the original control points to arrive at an estimated fingerprint sequence

$$\begin{cases} \tilde{\mathbf{w}}_x = \frac{\tilde{\mathbf{c}}_x - \mathbf{c}_x}{\alpha} \\ \tilde{\mathbf{w}}_y = \frac{\tilde{\mathbf{c}}_y - \mathbf{c}_y}{\alpha} \end{cases}. \tag{6}$$

We then evaluate the similarity between this estimated fingerprint sequence and each fingerprint sequence in our database through a correlation-based statistic.

In our work, we compute the correlation coefficient ρ and convert it to a Z-statistic by

$$Z = \log\left(\frac{1+\rho}{1-\rho}\right)\frac{\sqrt{2(n+1)-3}}{2}. \tag{7}$$

The Z-statistic has been shown to follow an approximate unit-variance Gaussian distribution with a large positive mean under the presence of a fingerprint, and a zero mean under the absence [14][15]. Thus, if the similarity is higher than a threshold (usually set between 3 to 6 for Z statistics), with high probability the corresponding fingerprint sequence in the database is present in the test curve, allowing us to trace the test curve to a specific user.

2.3 Fidelity and Robustness Considerations

Estimating the control points requires a set of properly chosen sample points from the curve. Uniform sampling can be used when there are no abrupt changes in a curve segment, while nonuniform sampling is desirable for curve segments that exhibit substantial variations in curvature.

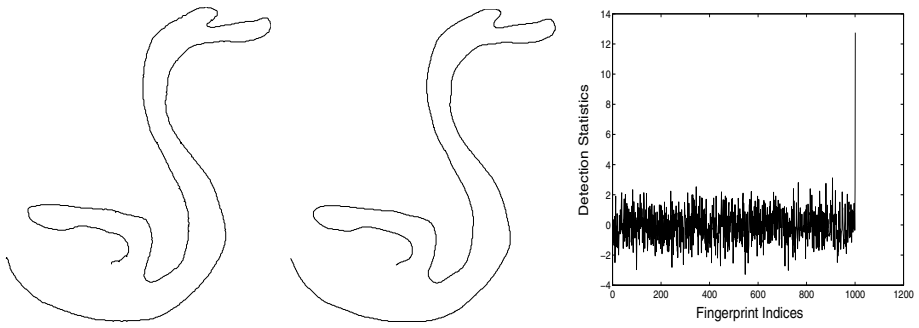
The number of control points is an important parameter for tuning. Depending on the shape of the curve, using too few control points could cause the details of the curve be lost, while using too many control points may lead to over fitting and bring artifacts even before data embedding. The number of control points not only affects the distortion introduced by the embedding, but also determines the fingerprint's robustness against noise and attacks. The more the control points, the longer the fingerprint sequence, and in turn the more robust the fingerprint against noise and attacks. In our tests, the number of control points is about 5-8% of the total number of curve pixels and the specific numbers will be provided with the experimental results.

The scaling factor α also affects the invisibility and robustness of the fingerprint. The larger the scaling factor, the more robust the fingerprint, but the larger the distortion resulted in. For cartographic applications, industrial standards provide guidelines on the maximum allowable changes [5]. Perturbation of 2 to 3 pixels is usually considered acceptable. We use random number sequences with unit variance as fingerprints and set α to 0.5 in our tests.

2.4 Experimental Results of Fingerprinting Simple Curves

We first present the fingerprinting results on a simple "Swan" curve in Figure 1(a) to demonstrate our basic embedding and detection algorithms. The curve was hand-drawn on a TabletPC and stored as a binary image of size 392×329. We use the contour following algorithm in [16] to traverse the curve and obtain its vector representation. Then uniform sampling of a total of 1484 curve points and the quadratic B-spline blending function (order $k = 3$) are employed to estimate 100 control points. Then we mark these control points and construct a fingerprinted "Swan" curve as shown in Figure 1(b). By comparing it with the original curve in Figure 1(a), we can see that they are visually identical.

Based on the assumption and the detection method discussed in Section 2.2, we obtain the detection statistic of ***12.75*** when correlating the fingerprinted "Swan" curve with the true user's fingerprint sequence and very small statistic values when with innocent users, as shown in Figure 1(c). This suggests that the embedded fingerprint is identified with high confidence.



(a) (b) (c)

Fig. 1. Fingerprinting a hand-drawn "*Swan*" curve: (a) the original curve, (b) the fingerprinted curve, (c) Z statistics

3 Robust Fingerprint Detection

The set of test sample points $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ assumed in Section 2.2 is not always available to a detector, especially when a test curve undergoes geometric transformations (such as rotation, translation, and scaling), vector/raster conversion, and/or is scanned from a printed hard copy. There must be a pre-processing registration step preceding the basic fingerprint detection module to align the test curve with the original one. In order to improve the accuracy and efficiency of the registration, an automated registration is desirable over a manual registration. With the affine invariance property, B-splines have been utilized in a few existing curve alignment works. In a recent method employing a *super-curve* [17], two affine-related curves are superimposed with each other in a single frame and then this combined *super-curve* is fitted by a single B-spline. Through minimizing the B-spline fitting error, both transform parameters and control points of the fitting B-spline can be estimated simultaneously. Since neither integration nor differentiation is needed, this method is robust to noise and will serve as a building block in our work.

Another problem related to the previous assumption is the inherent non-uniqueness of B-spline control points, which refers to the fact that a curve can be well approximated by different sets of B-spline control points. With a different time assignment or a different set of sample points, we may induce a quite different set of control points that can still accurately describe the same curve. It is possible for the differences between two sets of unmarked control points to be much larger than the embedded fingerprint sequence. Therefore, if we cannot

find from a test curve a set of control points corresponding to the one used in the embedding, we may not be able to detect the fingerprint sequence. Considering the one-to-one relationship between sample points (including their time assignments $\{s_j\}$) and control points, we try to find the set of sample points on a test curve that corresponds to the set of sample points used in the embedding. We shall refer to this problem as the *point correspondence problem*. As we shall see, the non-uniqueness issue of B-spline control points can be addressed through finding the point correspondence.

3.1 Problem Formulation

We now formulate the curve registration and point correspondence problem in the context of fingerprint detection.

We use "View-I" to refer to the geometric setup of the original unmarked curve and "View-II" the setup of the test curve. Thus we can register the two curves by transforming the test curve from "View-II" to "View-I", or transforming the original curve from "View-I" to "View-II". We focus on registration under the affine transformations, which can represent combinations of scaling, rotation, translation, and shearing. These are common geometric transformations and can well model common scenarios in printing-and-scanning.

We call two points (x, y) and $(\tilde{x}, \tilde{y})$ *affine related* if

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_x^T \\ \mathbf{a}_y^T \\ 0 \;\; 0 \;\; 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{8}$$

where $\{a_{ij}\}$ are parameters representing the collective effect of scaling, rotation, translation, and shearing. These transform parameters can be represented by two column vectors $\mathbf{a}_x = [a_{11} \;\; a_{12} \;\; a_{13}]^T$ and $\mathbf{a}_y = [a_{21} \;\; a_{22} \;\; a_{23}]^T$ or by a single matrix $\mathbf{A}$. Similarly, the inverse transformation can be represented by

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{g}_x^T \\ \mathbf{g}_y^T \\ 0 \;\; 0 \;\; 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} \tag{9}$$

The original curve available to the detector in fingerprinting applications can be a raster curve or a vector curve. The detector also knows the original set of sample points $(\mathbf{p}_x, \mathbf{p}_y)$ that is used for estimating the set of control points upon which spread spectrum embedding is to be applied. In addition to possible affine transformations between the original and the test curve, the correct point correspondence information may not always be available, i.e., the set of test sample points $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ assumed in Section 2.2 is absent. This is especially the case after a fingerprinted curve goes through vector-raster conversions and/or printing-and-scanning. Under this situation, not only transform parameters for the curve alignment but also the point correspondence must be estimated in order to locate the fingerprinted control points successfully. The test curve can be a vector curve with sampled curve points $(\tilde{\mathbf{v}}_x, \tilde{\mathbf{v}}_y)$ or a raster curve with pixel

coordinates $(\tilde{\mathbf{r}}_x, \tilde{\mathbf{r}}_y)$. As a vector curve can be rendered as a raster curve through interpolation, we consider that the original and the test curve are represented in raster format and formulate the problem as:

Given an original raster curve with a set of sample points $(\mathbf{p}_x, \mathbf{p}_y)$ *and a test raster curve* $(\tilde{\mathbf{r}}_x, \tilde{\mathbf{r}}_y)$*, we register the test curve with the original curve and extract the control points of the test curve. Both transform parameters* $(\mathbf{a}_x, \mathbf{a}_y)$ *(or equivalently* $(\mathbf{g}_x, \mathbf{g}_y)$*) and a set of sample points on the test curve* $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ *corresponding to the one used in the fingerprint embedding must be found from the test raster curve.*

3.2 Iterative Alignment-Minimization (IAM) Algorithm

To align the test curve with the original curve and in the mean time identify the point correspondence of the sample points, we develop an Iterative Alignment-Minimization (IAM) algorithm. The IAM algorithm is an iterative algorithm consisting of three main steps. We first obtain an initial estimation of the test sample points. With the estimated point correspondence, we then perform "super" curve alignment to estimate both the transform parameters and the control points of the test curve. With the estimated transform parameters, we refine the estimation of point correspondence through a nearest-neighbor rule.

Step-1 Initial Estimation of Sample Points on the Test Curve: We initialize the sample points $(\tilde{\mathbf{p}}_x^{(1)}, \tilde{\mathbf{p}}_y^{(1)})$ on the test curve using the following simple estimator. Let N and $\tilde{N}$ be the number of points on the original raster curve and on the test raster curve, respectively. From the known indices of the original curve's $m+1$ sample points $\mathbf{J} = [j_0, j_1, j_2, \ldots, j_m]$, where $j_0 < j_1 < j_2 < \ldots < j_m$ are integers ranging from 0 to $N-1$, we can estimate indices of the test curve's $m+1$ sample points by $\tilde{\mathbf{J}} = round\left(\frac{\tilde{N}-1}{N-1} \cdot \mathbf{J}\right)$. Using this estimated index vector $\tilde{\mathbf{J}}$, we can identify the corresponding sample points from the test curve and take them as the initial estimate.

Step-2 Curve Alignment with the Estimated Sample Points: Given the estimated point correspondence and the corresponding sample points $(\tilde{\mathbf{p}}_x^{(i)}, \tilde{\mathbf{p}}_y^{(i)})$ for the test curve in the i^{th} iteration, we apply the curve alignment method in [17] to estimate the transform parameters and the control points of the test curve. More specifically, let the transform parameters from View-I (the original curve) to View-II (the test curve) be $(\mathbf{a}_x^{(i)}, \mathbf{a}_y^{(i)})$. The sample points on the test curve can be transformed back to View-I by $(\mathbf{g}_x^{(i)}, \mathbf{g}_y^{(i)})$. We then fit these transformed test sample points as well as the original sample points with a single B-spline curve (referred to as a "super-curve" in [17]) and search for both the transform parameters $(\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)})$ and the B-spline control points $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$ to minimize the fitting error

$$f(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)}, \hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)}) = \left\| \begin{bmatrix} \mathbf{B} \\ \mathbf{B} \end{bmatrix} \hat{\mathbf{c}}_x^{(i)} - \begin{bmatrix} \mathbf{p}_x \\ \tilde{\mathbf{P}}^{(i)} \hat{\mathbf{g}}_x^{(i)} \end{bmatrix} \right\|^2 + \left\| \begin{bmatrix} \mathbf{B} \\ \mathbf{B} \end{bmatrix} \hat{\mathbf{c}}_y^{(i)} - \begin{bmatrix} \mathbf{p}_y \\ \tilde{\mathbf{P}}^{(i)} \hat{\mathbf{g}}_y^{(i)} \end{bmatrix} \right\|^2, \quad (10)$$

where $\tilde{\mathbf{P}}^{(i)} \triangleq [\,\tilde{\mathbf{p}}_x^{(i)} \quad \tilde{\mathbf{p}}_y^{(i)} \quad \mathbf{1}\,]$ and $\mathbf{1}$ is a column vector with all $1's$. The partial derivatives of the fitting error function with respect to $\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)}, \hat{\mathbf{c}}_x^{(i)}$, and $\hat{\mathbf{c}}_y^{(i)}$ being zero is the necessary condition for the solution to this optimization problem. Thus we obtain an estimate of the transform parameters and the B-spline control points as:

$$\begin{cases} \hat{\mathbf{g}}_x^{(i)} = \mathbf{C}^{(i)}\mathbf{D}^{(i)}\mathbf{p}_x, \quad \hat{\mathbf{g}}_y^{(i)} = \mathbf{C}^{(i)}\mathbf{D}^{(i)}\mathbf{p}_y, \\ \hat{\mathbf{c}}_x^{(i)} = \mathbf{D}^{(i)}\mathbf{p}_x, \quad \hat{\mathbf{c}}_y^{(i)} = \mathbf{D}^{(i)}\mathbf{p}_y \end{cases},$$
$$\text{where} \quad \begin{cases} \mathbf{C}^{(i)} \triangleq \left(\tilde{\mathbf{P}}^{(i)T}\tilde{\mathbf{P}}^{(i)}\right)^{\dagger}\tilde{\mathbf{P}}^{(i)T}\mathbf{B} \\ \mathbf{D}^{(i)} \triangleq \left(2\mathbf{B}^T\mathbf{B} - \mathbf{B}^T\tilde{\mathbf{P}}^{(i)}\mathbf{C}^{(i)}\right)^{\dagger}\mathbf{B}_0^T \end{cases}. \tag{11}$$

The estimated control points $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$ can then be used to estimate the embedded fingerprint sequence and further compute the detection statistic $Z^{(i)}$, as described in Section 2.2.

Step-3 Refinement of Sample Point Estimation on the Test Curve: Given the estimated transform parameters $(\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)})$, we align the test raster curve $(\tilde{\mathbf{r}}_x, \tilde{\mathbf{r}}_y)$ with the original curve by transforming it to View-I. As the fingerprinted sample points $(\mathbf{B}(\mathbf{c}_x + \alpha\mathbf{w}_x), \mathbf{B}(\mathbf{c}_y + \alpha\mathbf{w}_y))$ are located at the neighborhood of their corresponding unmarked version $(\mathbf{B}\mathbf{c}_x, \mathbf{B}\mathbf{c}_y)$, we apply a *nearest neighbor* rule to get a refined estimation of the test curve's sample points. More specifically, for each point of $(\mathbf{B}\mathbf{c}_x, \mathbf{B}\mathbf{c}_y)$, we find its closest point from the aligned test raster curve and then denote the collection of these closest points as $(\tilde{\mathbf{p}}_{x,I}^{(i+1)}, \tilde{\mathbf{p}}_{y,I}^{(i+1)})$. These nearest neighbors form refined estimates of the test sample points in View-I and are then transformed with parameters $(\hat{\mathbf{a}}_x^{(i)}, \hat{\mathbf{a}}_y^{(i)})$ back to View-II as new estimates of the test sample points:

$$\begin{cases} \tilde{\mathbf{p}}_x^{(i+1)} = \begin{bmatrix} \tilde{\mathbf{p}}_{x,I}^{(i+1)} & \tilde{\mathbf{p}}_{y,I}^{(i+1)} & \mathbf{1} \end{bmatrix} \hat{\mathbf{a}}_x^{(i)} \\ \tilde{\mathbf{p}}_y^{(i+1)} = \begin{bmatrix} \tilde{\mathbf{p}}_{x,I}^{(i+1)} & \tilde{\mathbf{p}}_{y,I}^{(i+1)} & \mathbf{1} \end{bmatrix} \hat{\mathbf{a}}_y^{(i)} \end{cases}. \tag{12}$$

After this update, we increase i and go back to Step-2. The iteration will continue until convergence or for an empirically determined number of times. A total of 15 rounds of iterations are used in our experiments.

3.3 Detection Example and Robustness Analysis

We present a detection example employing the proposed IAM algorithm on a curve taken from a topographic map. Shown in Figure 2(a) are the original curve and a fingerprinted curve undergone vector-raster conversion and some geometric transformations. After the vector-raster conversion, the point correspondence is no longer directly available from the curve representation and therefore our proposed IAM algorithm is desirable. The estimated sample points for the test curve after one iteration and 15 iterations are shown in Figure 2(b) and

Figure 2(c), respectively. We can see that the initial estimates deviate from the true values by a non-trivial amount, while after 15 iterations the estimated values converge to the true values. We plot the six estimated transform parameters for each iteration in Figure 2(d), which shows an accurate registration by the proposed IAM algorithm. Upon convergence, we utilize the estimated control points $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$ and arrive at a high fingerprint detection statistic value as shown in Figure 2(e). This suggests the positive identification of the correct fingerprint by using the proposed IAM algorithm.

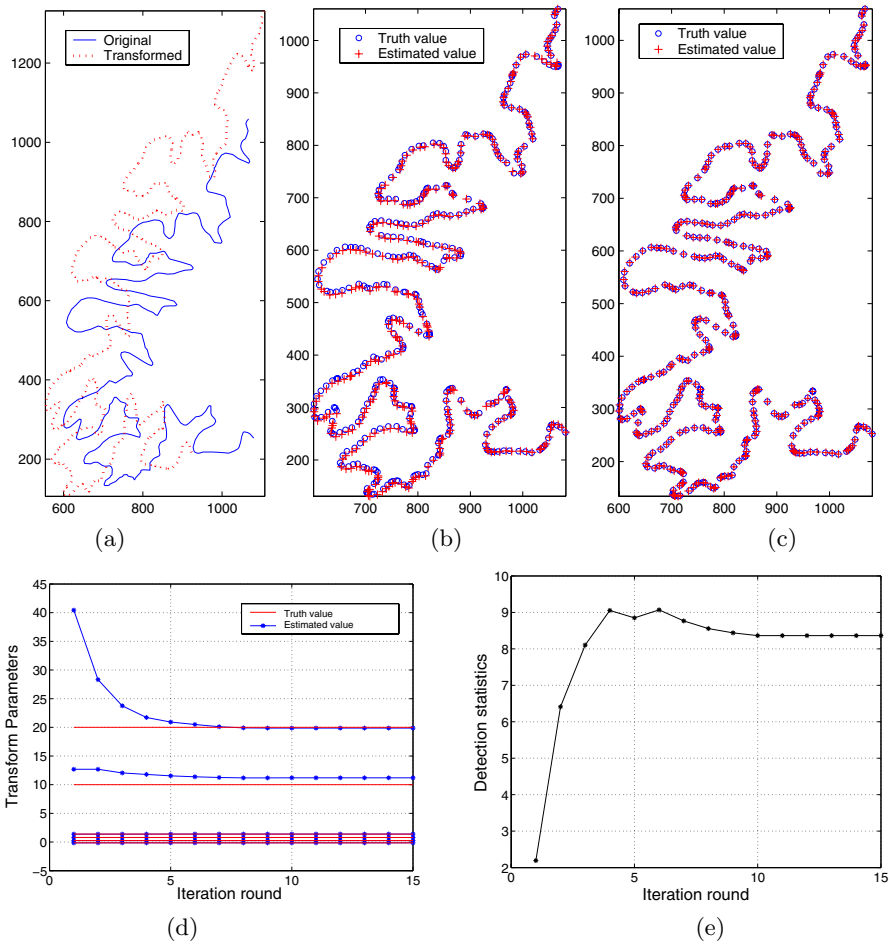


Fig. 2. Detection example using the IAM algorithm: a) the original and the test raster curve; b) estimated sample points after 1 iteration; c) estimated sample points after 15 iterations; d) estimated transform parameters; e) fingerprint detection statistic

The above example shows that through the IAM algorithm, we can register the test curve with the original unmarked curve and extract the fingerprinted

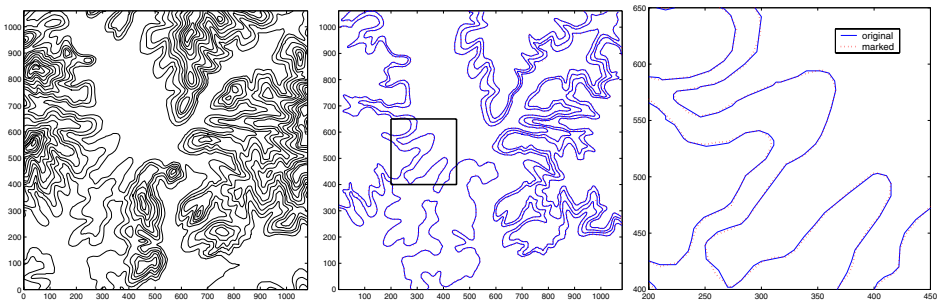
control points with high accuracy. With good estimation of affine transform parameters, our data embedding method for curves is resilient to combinations of scaling, rotation, translation, and shearing. The explicit estimation of point correspondence also provides resilience against the vector-raster conversion. With the robustness resulting from spread spectrum embedding in B-spline control points and the IAM algorithm, our curve fingerprinting approach can resist a number of challenging attacks and distortions. The next section will provide further demonstration.

4 Experimental Results for Map Fingerprinting

We now present experimental results of our curve fingerprinting approach in the context of tracing and tracking the topographic map, which provides a two-dimensional representation of the earth's three-dimensional surface. Vertical elevation is shown with contour lines (also known as level lines) to represent the earth's surfaces that are of equal altitude. Contour lines in topographic maps often exhibit a considerable amount of variations and irregularity, prompting the need of non-uniform sampling of curve points in the parametric modelling of the contours. In our experiments, the original map is stored in vector format. A set of discrete, non-uniformly vector points is defined for each contour line and used as sample points in our tests for estimating control points.

Fingerprinted Topographic Maps. A 1100×1100 topographic vector map obtained from http://www.ablesw.com is used in our experiment. Starting with the original map shown in Figure 3(a), we mark nine curves that are sufficiently long. A total of 1331 control points are used to carry the fingerprint. We overlay in Figure 3(b) these nine original and marked curves using solid lines and dotted lines, respectively. To help illustrate the fidelity of our method, we enlarge a portion of the overlaid image in Figure 3(c). We can see that the fingerprinted map preserves the geospatial information in the original map up to a high precision. The perturbation can be adapted to be compliant with cartographic industry standards and/or the need of specific applications.

Resilience to Collusion and Printing-and-Scanning. To show the robustness of our approach against the combinational attack of collusion and printing-and-scanning, we first generate a colluded map by averaging coordinates of the control points from four users' fingerprinted maps, then render it and print it out using a HP laser printer, and finally scan back as a binary image by a Canon scanner with 360dpi resolution. Preprocessing before detection includes a thinning operation to extract one-pixel wide skeletons from the scanned curves that are usually several-pixel wide after high resolution scanning. By using the proposed IAM algorithm, we get Z statistics of ***7.27, 8.91, 6.15, 8.12*** for the four colluders, indicating that the embedded fingerprints from all the four colluders survive this combinational attack thus the sources of leak for this map can be identified. This combinational attack also involves vector-raster conversion and affine transformations, and the result shows the resilience of our method to them.



(a) (b) (c)

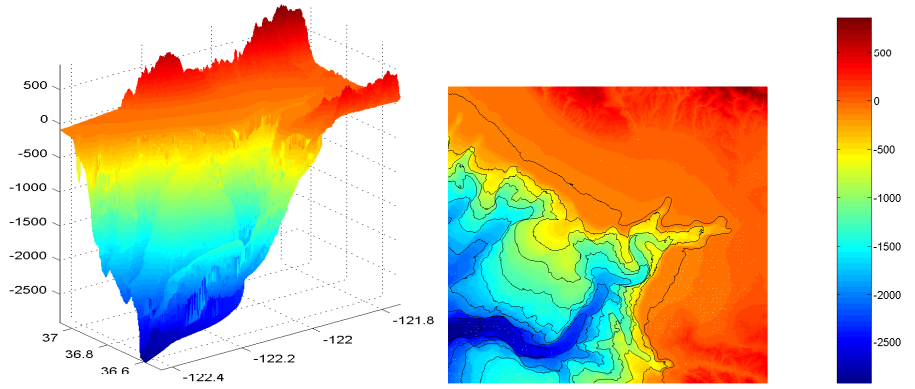
Fig. 3. Fingerprinting topographic maps: (a) original map, (b) original and fingerprinted curves overlaid with each other; (c) a zoomed-in view of (b)

Resilience to Point Deletion in Vector and Raster Maps. As traitor tracing applications usually involve adversaries who have strong incentives to remove the fingerprints, attackers may delete a certain number of points from a fingerprinted vector/raster map while keeping similar shapes of its contour lines. For a fingerprinted vector map, 20% points are randomly chosen and removed from each fingerprinted curve, while in a fingerprinted raster map 70% points are randomly chosen and removed. As the point correspondence is corrupted by the point deletion, we first construct a raster map from the remaining points by linear interpolation and then apply our IAM algorithm. The detection statistics for these two cases are ***11.40*** and ***15.61***, respectively. Thus the embedded fingerprints survive point deletion applied to both vector maps and raster maps.

5 Extension to Fingerprinting 3D Geospatial Data

In addition to 2D topographic maps, geospatial data are often acquired and archived as a 3D data set, which includes a set of spatial locations and their height information. The 3D geospatial data can be represented both as 3D surface and as 2D contours. An example of Monterey Bay region is shown in Figure 4, where the 3D oceanfloor depth data obtained from the U.S. National Geophysical Data Center (NGDC) [18] are rendered in Figure 4(a), and the corresponding 2D topographic map is shown in Figure 4(b). Since the same geospatial data set can be represented using different dimensionalities, we explore in this section such fingerprinting method that can allow the fingerprints to be detected from both the 3D data set and the 2D representation, whichever readily available to the detector. This can be achieved by extending our proposed fingerprinting method for curves.

Our basic idea of fingerprinting 3D elevation data is as follows. Given a set of 3D elevation data, we first extract its 2D contours and then embed a watermark into these contour curves using the method presented earlier in this paper. From the original 3D elevation data set and the marked 2D contours, we construct a marked 3D elevation data set. The new issues to be addressed here are how



(a) (b)

Fig. 4. Elevation map: (a) a 3D geospatial data set, (b) the corresponding 2D contours

to perform 2D contour extraction from a 3D data set and how to construct a marked 3D elevation data set from marked 2D contours.

The 2D contour extraction starts from performing a planar cut of the elevation data set at a selected height and then a binary map can be generated by assigning $1's$ to locations of height greater than the given height and $0's$ to the other locations. We apply a robust edge detector, such as the Canny method, to the binary map to obtain a raster representation of the contour of the given height. Using the contour following algorithm in [16], we traverse this raster contour and obtain a vector representation. Non-uniform sampling is employed to acquire more samples for segments with substantial variations in curvature and fewer in flat areas. Finally, we use the least-squares approach to estimate the B-spline control points of the contour.

When constructing a marked 3D data set, we need to generate a set of 3D elevation data under application-specific constraints, including preserving the geospatial meanings and ensuring the marked 2D contours can be estimated from the marked 3D data set with high accuracy. We recall that in the embedding stage, each original 2D contour is generated from a binary image with $1's$ in locations of height greater than the contour height and $0's$ in other locations. As the marked 2D contour represents a slightly different partition, we need to modify the height values of the 3D data set for locations around the marked contour so that the 2D contour generated from this new 3D data set is the same as the marked 2D contour. To accomplish this, we search for the locations that have higher elevation in the marked 2D contour map than in the original one, and modify their height in the 3D data set to be slightly higher. Similar adjustment is made to the locations with lower elevation in the marked 2D contour map than before.

To extract fingerprint from a 3D elevation data set, we perform some basic alignment if necessary. Using the same contour extraction approach as in the embedding, we then extract from the 3D data set the 2D contours for the same

elevations as selected in the embedding. Since the extracted contours are in raster format, we can apply the IAM algorithm as discussed earlier in Section 3 to extract the embedded fingerprint.

We apply the proposed fingerprinting method to the Monterey Bay data set of Figure 4(a). We hide a fingerprint sequence in three 2D contours of height -100, -400, and -700, respectively. A total of 484 control points are used for carrying the fingerprint. To detect fingerprint, we extract the three 2D contours for the above heights from the fingerprinted 3D data set and obtain a Z detection statistic of ***8.02*** through the proposed IAM algorithm. This suggests a successful fingerprint detection from these three elevation levels.

6 Conclusions

In this paper, we have presented a new data hiding algorithm for curves by parameterizing a curve using the B-spline model and adding spread spectrum sequences to curve parameters. In conjunction with the basic embedding and detection techniques, we have proposed an iterative alignment-minimization algorithm to allow for robust fingerprint detection under unknown geometric transformations and in absence of explicit point correspondence. We have demonstrated the fidelity of our method as well as its robustness against collusion, affine transformations, vector-raster conversion, printing-and-scanning, and their combinations. Our work has shown the feasibility of the proposed algorithm in fingerprinting applications for tracing and tracking topographic maps as well as writings/drawings from pen-based inputs. We have also extended our curve watermarking method from protecting 2D topographic maps to 3D elevation data sets. The protection of all of these documents has increasing importance to the emerging digital operations in government, military, intelligence, and commerce.

References

1. Wu, M., Trappe, W., Wang, Z., Liu, K.J.R.: Collusion resistant fingerprinting for multimedia. IEEE Signal Processing Magazine **21** (2004) 15–27.
2. Chang, H., Chen T., Kan K.: Watermarking 2D/3D graphics for copyright protection. Proc. of IEEE ICASSP (2003) 720–723.
3. Barni, M., Bartolini, F., Piva, A., Salucco, F.: Robust watermarking of cartographic images. EURASIP Journal on Applied Signal Processing **2** (2002) 197–208.
4. Solachidis, V., Pitas, I.: Watermarking polygonal lines using fourier descriptors. IEEE Computer Graphics and Applications **24** (2004) 44–51.
5. Ohbuchi, R., Ueda, H., Endoh, S.: Watermarking 2D vector maps in the mesh-spectral domain. Proc. of the Shape Modeling International (SMI) (2003).
6. Koch, E., Zhao, J.: Embedding robust labels into images for copyright protection. Proc. Int. Congr. Intellectural Property Rights for Specicalized Information, Knowledge and New Technologies (1995).
7. Wu, M., Tang, E., Liu, B.: Data hiding in digital binary image. Proc. of IEEE ICME (2000) 393–396.

8. Wu, M., Liu, B.: Data hiding in binary image for authentication and annotation. IEEE Trans. on Multimedia Vol.**6** (2004) 528–538.
9. Ohbuchi, R, Masuda, H., Aono, M.: A Shape-Preserving Data Embedding Algorithm for NURBS Curves and Surfaces. Proc. of CGI (1999) 180–187.
10. Lee, J. J., Cho, N. I., Kim, J. W.: Watermarking for 3D NURBS graphic data. Proc. of IEEE MMSP (2002) 304–307.
11. Cox, I., Killian, J., Leighton, F., Shamoon, T.: Secure spread spectrum watermarking for multimedia. IEEE Transcations on Image Processing **6** (1997) 1673–1687.
12. Jain, A.K.: Fundamentals of Digital Image Processing. Prentice Hall (1989).
13. Farin, G.E.: Curves and surfaces for computer-aided geometric design: a practical guide. 4th edn. Academic Press (1997).
14. Stone, H.: Analysis of attacks on image watermarks with randomized coefficients. Technical Report 96-045, NEC Research Institute (1996).
15. Zhao, H., Wu, M., Wang, Z., Liu, K.J.R.: Nonlinear collusion attacks on independent multimedia fingerprints. accepted by IEEE Trans. on Image Proc (2004).
16. Cabrelli, C., Molter, U.: Automatic representation of binary images. IEEE Trans. on Pattern Analysis and Machine Intelligence **12** (1990) 1190–1196.
17. Xia, M., Liu, B.: Image registration by 'super-curves'. IEEE Trans. on Image Processing **13** (2004) 720–732.
18. National Geophysical Data Center (NGDC): http://www.ngdc.noaa.gov, U.S. Oceanic and Atomospheric Administration.