A High Capacity 3D Steganography Algorithm

Min-Wen Chao, Chao-hung Lin, Cheng-Wei Yu, and Tong-Yee Lee, Member, IEEE

Abstract—In this paper, we present a very high-capacity and low-distortion 3D steganography scheme. Our steganography approach is based on a novel multilayered embedding scheme to hide secret messages in the vertices of 3D polygon models. Experimental results show that the cover model distortion is very small as the number of hiding layers ranges from 7 to 13 layers. To the best of our knowledge, this novel approach can provide much higher hiding capacity than other state-of-the-art approaches, while obeying the low distortion and security basic requirements for steganography on 3D models.

Index Terms—Information hiding, 3D steganography, multilayered embedding.

1 INTRODUCTION

STEGANOGRAPHY is a technique to hide secret messages in a host media called cover media. The advantage of steganography over cryptography is that messages do not attract attention to attackers and even receivers. Steganography and cryptography are often used together to ensure security of the secret messages [33]. For example, many previous steganography approaches [27], [28], [29], [34] use the secret key (i.e., idea borrowed from cryptography) to produce better protection of the information if the stego object arouses suspicion. Therefore, these approaches can become more secure and can be potentially useful to some security-demanding applications such as military intelligence. Watermarking is another popular technique to hide messages and it is usually used for providing ownership on copyrighted multimedia material and for detecting originators of illegally made copies. Therefore, an effective watermarking method must be robust against a variety of attacks. In contrast to watermarking, steganography prefers to hide information as much as possible and requires cover media with distortion as little as possible [1], [2].

In the past, most of the effort on steganography has been concentrated on various media data types such as an image, an audio file, or even a video file. However, there is relatively little steganography work on 3D models. With the development of 3D hardware, 3D computing or visualization has become much more efficient than ever. This leads to the widespread use of 3D models in various applications such as digital archives, entertainment, Web3D, MPEG4, and game industry. Therefore, 3D models are good candidates and rich resources to serve as the innocuous-looking hosts for hiding other types of digital content.

There have been many good 3D watermarking methods proposed [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25]. Most of them may be easily modified for the purpose of steganography. However, the data-hiding capacity can be very low, since they are not originally designed for this purpose. In this paper, we will exploit the geometric characteristics of 3D models to provide high-capacity data hiding. Usually, there is a trade-off between capacity, invisibility, and robustness in steganography techniques [3]. Capacity and invisibility are more important than robustness in the steganography system [1], [2], [3]. Therefore, we aim at maximizing data-hiding capacity while limiting distortion of cover models in a lower bounded value.

The major contributions of this paper are listed below:

- To the best of our knowledge, our novel approach can hide much higher bit rates/vertex (i.e., 21 to 39 bits) than other previous state-of-the-art methods in steganography for 3D polygon models.
- The vertex distortion of a covert model can be limited to an upper bound \( I/2 \), where \( I \) is the width of a partitioning interval. The capacity and distortion of our approach depends mainly on the number of partitioning intervals. We optimize the number of partitioning intervals in terms of capacity and distortion.

2 RELATED WORKS

We describe only previous information hiding efforts for 3D polygon models. As for other cover media such as documents, images, video, and audio, a brief history and surveys can be found in [1] and [26]. Information hiding encompasses a broad range of applications in which the messages are embedded into the cover media for a variety of purposes. Watermarking and steganography are two types of information hiding. A significant number of methods have been proposed for watermarking [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25]. The common purpose of watermarking is to provide a robust algorithm to withstand various malicious attacks for digital content protection or to locate the modification for authenticating the integrity of...
digital content. In general, there is a strong relationship between the embedded digital watermark and the host content. In contrast, the hidden information in steganography is kept secret to the public. Therefore, watermarking techniques are generally not suitable for the covert communication. Among these watermarking techniques, [7] and [15] could have better chances of application to steganography. After some proper modifications, [15] can hide about 0.5–1.0 bit per vertex and [7] can hide 2 bits per vertex. However, the performance of these two methods, in terms of capacity, is still much less than our approach. Instead of focusing on the robustness issue as done in most digital image watermarking research, we take capacity, reliability, and security into serious consideration. The goal is to maximize capacity while ensuring that the amount of hidden information can be transmitted without errors.

3D model watermarking technology has been actively studied in the past, but only a few works have addressed information hiding for covert communication [27], [28], [29]. In [27], the authors presented a blind information hiding scheme based on a substitution procedure in the spatial domain. This scheme is derived from the well-known quantization index modulation (QIM) [30], [31]. The basic idea is to represent a triangle as a two-state, i.e., state “0” or “1,” geometrical object depending on what bit value is to be hidden. They adopted a technique called the triangle strip peeling sequence (TSPS) presented in [10] and [11] to determine the data embedding order for security considerations. In [28], the authors introduced a multilevel embedding procedure extending the scheme proposed in [27] for expanding the hiding capacity. They propose three embedding levels called sliding, extending, and rotating to embed data based on slightly shifting the vertex position. This method can provide about three times the capacity of that in [27]. In [29], the authors extended their previous work [28]. In addition to a multilevel embedding procedure, they proposed a 3D model representation rearrangement procedure to hide more data (9 bits/vertex). In contrast, our novel multilayered embedding scheme can offer up to $3n_{\text{layers}}$ bits/vertex, where $n_{\text{layers}}$ ranged from 7 to 13 in our experiments. Our approach produces much higher hiding capacity than all of the previous works.

3 OVERVIEW OF STEGANOGRAPHIC SYSTEM

Fig. 1 schematically illustrates the proposed information hiding scheme. This scheme consists of two separate procedures: the embedding procedure and the extraction procedure. Both procedures have two main steps: spatial analysis and multilayered embedding/extraction. We briefly describe the embedding procedure. In the spatial analysis step, we analyze the cover model for obtaining a vertex embedding order and three end vertices for the payload embedding. The vertex traverse approach presented in [27] and [28] is adopted to generate the vertex embedding order. The basic idea is to represent a triangle as a pivot edge and two possible exit edges. The traverse order starts with an initial triangle and the next visited triangle is determined by the bit value in a selected secret key to take security into account. The initial triangle is determined by the principal component analysis (PCA) of the cover model. Given the first principal axis obtained from the PCA for the cover model, the initial triangle is the one that intersects this axis and its position is furthest from the cover model center of gravity. Once the initial triangle is determined, the next visited triangle is selected depending on the next bit value in the secret key. If the next bit value is “1,” the traversal direction is the right exit edge. If the next bit value is “0,” the traversal direction is the left exit edge. Once the vertex embedding order is determined, the next step is to embed the payload on the vertices in that order. In this paper, a novel multilayered embedding scheme is proposed for enlarging the hiding capacity. In the extraction procedure, the embedding order can also be obtained using the secret key in the spatial analysis step. The payload can then be correctly extracted in the reverse embedding order.

4 MULTILAYERED EMBEDDING AND EXTRACTION SCHEMES

In this section, we describe the single-layered embedding scheme first and then detail the multilayered version.

4.1 Single-Layered Embedding Scheme

First, three vertices of a cover model denoted as $V_a$, $V_b$, and $V_c$ are selected. They are also called end vertices in this paper. These three end vertices are determined using the PCA of the cover model. Given the first and second principal axes, we orthogonally project all vertices onto these two axes. The vertices that fall on the two extreme ends of the first principal axis are selected as the end vertices $V_a$ and $V_b$. The vertex that fall on the furthest extreme end of the second principal axis is selected as the third end vertex $V_c$. If an end vertex has multiple candidates, we simply select the nearest candidate (nearest to the principle axis) as the end vertex and slightly shift the other candidates in order to uniquely define an end vertex. The next step is to transform the cover model to align the vectors $V_aV_b$ and $(V_aV_b \times V_aV_c)$ with the $x$-axis and $y$-axis.
regions. In this example, the following two cases:

interleaved manner (e.g., 010101 ...), denoted as $R_0$ and $R_1$ in Fig. 2. The suffix of $R_0$ and $R_1$ represents the binary feature states “0” and “1,” respectively. The region in which a vertex $v_i$ falls into can be simply calculated using

$$d_i/I = q_i; \ldots; r_i,$$

$$q_i = [d_i/I],$$

$$r_i = d_i \% I,$$

where $d_i$ represents the projected distance between vertex $v_i$ and the end vertex $V_a$ on the $x$-axis, that is, $d_i = v_i - V_a$ (where $v_i$ and $V_a$ represent the $x$ component of $v_i$ and $V_a$, respectively). $I$ represents the interval width of a state region, that is, $I = D/n_{intervals}$ where $n_{intervals}$ is the number of state regions and $D$ is the distance of $V_aV_b$. $q_i$ then represents the state region where a vertex $v_i$ falls into and $r_i$ represents the vertex position in the state region $q_i$, as shown in Fig. 2. “$\%$” is the remainder operator.

If $q_i \in R_0$, it implies that the bit value hidden in a vertex $v_i$ is “0.” Similarly, if $q_i \in R_1$, it implies that the bit value hidden in this vertex is “1.” However, the bit value that will be embedded in a vertex depends on the hidden data and the vertex embedding order. A vertex falls in the region $q_i \in R_0$ but the hidden bit value may be “1.” To solve this problem, we further divide each state region $q_i$ into two subregions, called the change region and the unchange region, as shown in Fig. 2. The purpose of the state region subdivision is to find an empty region, i.e., the change region (no vertex falls in), over all state regions in order to proportionally move the mismatched vertices ($q_i \in R_0$ but the hidden bit value is not equal to $k$, where $k = 0$ or 1) to the empty region. In this manner, we not only can hide the correct bit value in each vertex but have the ability to reverse the vertex position when the hidden bit value is extracted. These two subregions can be determined as follows: two subregions can be determined as

Case 1. $q_i \in R_0$ and $bit(i) = k, k = 0$ or 1: No modification.
Case 2. $q_i \in R_k$ and $bit(i) \neq k, k = 0$ or 1: Shift vertex $v_i$ to the neighboring change region in $q_i$, i.e., if $d_i \% I \leq I/2$,

move $v_i$ to the change region $[0, n]$; otherwise, move $v_i$ to the change region $[I - n, I]$, as shown in Fig. 3.

In the cases above, $bit(i)$ represents the bit value of payload hidden in a vertex $v_i$. In Case 2, the new position is calculated by adding a small offset $\delta_i$ to $d_i$ (as shown in Fig. 3), that is,

$$d'_i = \begin{cases} d_i - \delta_i, & \text{if } r_i \leq I/2, \\ d_i + \delta_i, & \text{if } r_i > I/2, \end{cases}$$

where $\delta_i = \min(r_i, I - r_i)/(1 - 2n/I)$.

The drawback of the above hiding method is that, i.e., the change region, might be too small to precisely hide a payload. We propose two approaches to solve this problem. The first approach is to optimize the number of state regions $n_{intervals}$ (Section 4.6). The goal of this optimization is to enlarge the change regions while reducing the distortion. Another approach is to enlarge the change regions by simply shrinking the unchange region to the range $[I/4, 3I/4]$ and setting the change regions to be the regions $[0, I/4]$ and $[3I/4, I]$, as shown in Fig. 4. In this manner, the small offset $\delta_i$ in (2) must be redefined as

$$\delta_i = \min(r_i, I - r_i)/2.$$  (3)

Therefore, in Fig. 4, when embedding 1 bit $bit(i)$ to the vertex $v_i$, the new position $d'_i$ of this vertex can be calculated using

$$d'_i = \begin{cases} d_i + (I/4 - r_i)/2, & \text{if } q_i \in R_k, bit(i) = k, \\ d_i - \text{sgn}(I/2 - r_i)\delta_i, & \text{if } q_i \in R_k, bit(i) \neq k, \end{cases}$$

where $\text{sgn} x = \begin{cases} -1, & \text{if } x < 0, \\ 1, & \text{if } x \geq 0. \end{cases}$

In general, the offset $|d - d'|$ must be small enough to avoid visual degradation of the cover model but large enough to allow accurate payload detection. This will be discussed in detail in Sections 4.4-4.6.
is coded by color ranging from blue (the smallest distortion: 0.0) to red (the largest distortion: $5 \times 10^{-4}$).

Fig. 6. An illustration of the multilayered information hiding. The hiding layers number up to 10. The distortion (measured by root-mean-square error) for each vertex $v_i$ is given by $d_i$, and the bit value $bit(i)$ is calculated using (6).

In this manner, we can embed data on $x$, $y$, $z$-components of a vertex coordinate, respectively. In each layer, every vertex can hide 3 bits of information, except for the end vertices $V_{a}$, $V_{b}$, and $V_{c}$. These three vertices cannot be embedded with any messages because they are the bases for the extraction process. As a result, the theoretical upper bounded capacity of our hiding scheme is $3(|V| - 3)n_{layers}$, where $n_{layers}$ represents the number of embedding layers and $|V|$ represents the number of vertices. Each vertex swings in a 1/2 interval using the proposed multilayered architecture. Therefore, the maximal distortion is limited to 1/2. In Fig. 6, we show an example of our multilayered information hiding. The hiding layers number up to 10. The distortion is still imperceptible.

Note that the end vertices and initial triangle (mentioned in Section 3), obtained from cover and stego models, respectively, might not be identical if the distortion of the stego model is too large. Fortunately, this case had been very rare since the offset of vertex position is extremely small. In our experiments, the PSNR rates are all above 81 dB (shown in Tables 1 and 2). To test for robustness against this problem, we recheck if the end vertices and initial triangle are identical after data embedding.

4.3 Multilayered Extraction Scheme

In the multilayered extraction processes, the payloads are extracted in a reverse embedding order, i.e., from the last layer to the first layer, that is, extracting the bit values in reverse vertex embedding order. First, the variables $d_i$, $q_i$, and $r_i$ for each vertex $v_i$ can be calculated using (1) and (5). We then detect in which region (change or unchange region) the vertex lies using the following decision equation:

$$
\begin{align*}
    v_i &\in \text{change region}, & \text{if } r_i < I/4 \text{ or } r_i > 3I/4, \\
    v_i &\in \text{unchange region}, & \text{if } r_i \geq I/4 \text{ or } r_i \leq 3I/4.
\end{align*}
$$

The bit value $bit(i)$ embedded on the vertex $v_i$ can be extracted by checking its state region:

$$
\begin{align*}
    bit(i) &= \begin{cases} 
        q_i/2, & \text{if } v_i \in \text{unchange region,} \\
        1 - q_i/2, & \text{if } v_i \in \text{change region.}
    \end{cases}
\end{align*}
$$

To recover the vertex position in the previous layer, we subtract the small offset $d_i$ back from the variable $d_i$ [refer to (4)] and then expand the state region from $[I/4, 3I/4]$ to $[0, I]$, as shown in Fig. 7. The vertex position in the previous layer can then be calculated using the following equation, i.e., proportionally moving back to the previous layer:

$$
\begin{align*}
    d_i' &= d_i + q_i r_i - \frac{2}{3} d_i, \\
    r_i' &= r_i - q_i.
\end{align*}
$$
\[ d_i = \begin{cases} d_i - (I/2 - r_i), & \text{if } d_i \in \text{unchange region}, \\ d_i + \text{sgn}(I/2 - r_i) r_i, & \text{if } d_i \in \text{change region}, \end{cases} \quad (8) \]

where \( r_i = \min(r_i, I - r_i) \) and sgn is a sign function as defined in (4).

The overall multilayered extraction algorithm is processed as follows:

1. Calculate the variable \( d_i \) for each vertex \( v_i \).
2. Calculate the variable \( r_i \) for each vertex using (5).
3. Extract a bit value from \( v_i \) using (7).
4. Recover the vertex position on the previous layer using (8).
5. Repeat steps 1-4 for the previous layer extraction until all payloads are completely extracted.

### 4.4 Hiding Capacity Analysis

In our experiment, the vertex coordinate is represented by IEEE 754 single precision standard format, which occupies 32 bits and has a significant precision of 23 bits (about seven decimal digits). The proposed approach hides data in the spatial domain. Therefore, the scale of 3D model will affect the hiding capacity. To faithfully measure the hiding capacity of the proposed approach, the vertex coordinates of all the experimental models are normalized to the range of 0.0 to 1.0 before data is embedded.

In our multilayer data hiding scheme, the boundaries of change and unchange subregions in a state region are at \( I/4 \) and \( 3I/4 \) in the first layer. These two boundaries are shifted to \( 3I/8 \) and \( 5I/8 \) in the second layer due to space modulation, i.e., enlargement of the change region. Induct this to the \( n \)th layer, and these two boundaries will lie at \((2^n - 1)I/2^{n+1}\) and \((2^n + 1)I/2^{n+1}\). It means that the distance of any two adjacent boundaries in the \( n \)th embedding layer is \( I/2^n \). The smallest unit (precision) of a single floating point is \( 2^{-23} \). If the distance between these two boundaries is shorter than \( 2^{-23} \), we cannot hide any bit in this area. Therefore, \( I/2^n \) must be greater than or equal to \( 2^{-23} \), and the limitation of the number of layers can be formulated as

\[ n_{\text{layers}} \leq \log_2 I + 23. \quad (9) \]

From (9), we can see that the theoretically maximum number of hiding layers (limit) is 23 in the proposed multilayered embedding scheme. It occurs when the interval width of state region \( I \) is set to 1. It means that the number of state regions is only one, since the model is normalized to the range \([0.0, 1.0]\) [refer to (1)]. Therefore, the theoretical hiding capacity is 69 bits/vertex (each vertex can hide 3 bits in every layer).

### 4.5 Floating Point Rounding Scheme

In general, we cannot hide payload up to the theoretically maximum layers due to the precision rounding error. In the IEEE 754, if a number cannot be represented in the floating point format, then the float conversion requires a choice of which float-point number is appropriate. In the IEEE 754, the default method is the round-to-even approach, which is also sometimes called banker rounding. This approach chooses the nearest value. In our embedding scheme, this rounding scheme will cause wrong bit due to the shift of the vertex to the adjacent region (wrong region). This occurs when the vertex is very close to the region boundary. Take Fig. 8 as an example. The \( d_i^e \) for vertex \( v_i \) lies in the region \( R_b \) after payload embedding. The value \( d_i^e \) cannot be represented in the floating point format. Therefore, it is rounded...
Fig. 8. Floating point rounding. Case a: Rounded to the nearest value. Case b: Rounded to the nearest value within the same region. (The red lines represent the smallest unit in standard single precision format; the black lines represent the region boundary.)

to the nearest value (case a). However, this rounding process shifts \( d_i^* \) from region \( R_b \) to region \( R_c \). This will cause wrong payload embedding. To solve this problem, we slightly modify the rounding scheme by rounding \( d_i^* \) to the nearest value within the same region (as shown in case b). In this manner, we can ensure that the vertex falls in the correct region.

4.6 Interval Number Optimization

The capacity, i.e., the number of layers, and distortion mainly depend on the number of partitioning intervals in the proposed approach. A small partitioning interval can obtain a small distortion result but might hide less data, i.e., the change region is small. To balance the capacity and distortion, we find the optimal number of partitioning intervals by minimizing an objective function:

\[
F(n_{\text{intervals}}) = \alpha \times \text{Distortion} + (1-\alpha) \times \text{Capacity},
\]

where the instance \( n_{\text{intervals}} \) represents the number of partitioning intervals, and \( \alpha \) is a user-defined weight coefficient. In our experiments, the parameter \( \alpha \) is set to 0.5. Both \( \text{Distortion} \) and \( \text{Capacity} \) terms are explained in the following.

The \( \text{Distortion} \) term is simply formulated as the root-mean-square error (RMSE). The error is defined as the distance between the vertex \( v_i \) in the cover model and its corresponding vertex \( v_i^* \) in the stego model. It can also be calculated as the projected distances on the \( x, y \), \( z \)-components of the vertex coordinate, that is, \( (d_{ij} - d_{ij}^*) \), where \( 1 \leq j \leq 3 \). The worst case of a vertex distortion occurs when a vertex is shifted to the boundaries or the center of a state region. Therefore, we formulate this term as

\[
\text{Distortion} = \frac{1}{|V|} \sum_{i,j=1}^{3} (v_i - v_i^*)^2 = \frac{1}{3|V|} \sum_{i,j=1}^{3} (d_{ij} - d_{ij}^*)^2
\]

(11)

The \( \text{Capacity} \) term is estimated as the width of the unchanged region. A small unchanged region indicates that we can have a high probability (i.e., space) to swing vertices in the half interval of a state region. This implies that it might hide more layers. We adopt the standard deviation, denoted as \( \sigma \), as the estimator of this term. The confidence interval is set to be 1.96\( \sigma \). It means that about 95 percent vertices have fallen in this interval for a normal distribution of vertex data. Therefore, the width of an unchanged region is approximated as \( 2 \times 1.96\sigma \) (two sides) plus the distance between \( I/4 \) and the mean value of variable \( r \), as shown in Fig. 9, that is,

\[
\bar{r} = \frac{1}{3|V|} \sum_{i,j=1}^{3} r_{ij},
\]

\[
\sigma = \frac{1}{3|V|} \sum_{i,j=1}^{3} \sqrt{(r_{ij} - \bar{r})^2},
\]

(12)

\[\text{Capacity} = 2 \times 1.96\sigma + |I/4 - \bar{r}|.\]

Any efficient algorithm for solving the optimization problem can be adopted here. In this paper, we adopt the gradient-descent algorithm and the search range is set to \([|V|/2, 3|V|/2]\). The search space is not too large. Therefore, this process will not be time consuming. To avoid artifacts such as face fold-over in an extreme case where the interval \( I \) is larger than the half length of the smallest edge, we simply restrict its search range to be larger than \( 2D/|e| \), where \(|e|\) represents the length of the smallest edge and the parameter \( D \) is defined previously in Section 4.1.

5 Experimental Results

To validate the feasibility of the proposed approach, various 3D models have been selected in the experiments, as shown in Table 1 and Fig. 10. We adopt the commonly used peak signal-to-noise ratio (PSNR) to measure the distortion. Since the connectivities of the cover and stego models are identical, PSNR can be easily calculated via the RMSE of these two models. The RMSE is defined as \( \sqrt{\frac{1}{|V|} \sum_{i} \|v_i - v_i^*\|^2} \) and \( \sqrt{\frac{1}{|V|} \sum_{i} \|n_i - n_i^*\|^2} \) in the position and normal distortion measurements, respectively, and PSNR is defined as \( 20 \log_{10} \left( \frac{D_{\text{MAX}}}{\text{RMSE}} \right) \) (where \( D_{\text{MAX}} \) is set to the diagonal distance of the bounding box of the cover model and 180 degree in the position and normal distortion measurements, respectively).

The experimental results show that the number of layers \( n_{\text{layers}} \) ranges from 7 to 13 and the PSNR for position and normal distortions are all above 82 dB for the test models. Our approach can hide very high volumes of data, while the incurred distortion is extreme small. Note that the number of layers and the volume of embedding data shown in Table 1 are not the capacity limit of the proposed approach. As mentioned in Section 4, the layer limit is up to 23. The number of layers shown in Table 1 is the obtained optimal value for...
balancing the capacity and distortion. We show the analysis of the relation between capacity (the number of layers) and distortion (PSNR) in Fig. 11. We see that PSNR rates exponentially decrease with the number of layers. The PSNR decreases sharply when the number of layers is larger than 10 (the obtained optimal value, see Table 1) in the example of a horse model. This implies that we can find a suitable \( n_{\text{intervals}} \) in the optimization process. In Fig. 11, we also show the worst embedding case (the blue curve), in terms of distortion. The worst case scenario is to shift every vertex right or left by half of an interval \( I/2 \) in the embedding domain. Therefore, the position distortion is \( 20 \times \log_{10} \left( \frac{D_{\text{max}}}{\sqrt{I/2}} \right) \) dB. In the case of \( n_{\text{intervals}} = 8,385 \), i.e., the obtained optimal partition interval number, the PSNR is still up to 87 dB. It implies that the difference between the stego and the cover models is imperceptible even for the worst embedding case. We also
present the embedding analysis using the double precision standard format in Fig. 11b. The number of layers can be increased up to about 40. To further demonstrate the imperceptibility with respect to the human visual system, we synchronously magnify the local regions of the cover and stego models to perceive the detailed difference between them in Figs. 12 and 13. We also visualize the distortion encoding by colors in Fig. 14.

We experimentally evaluate our approach with two extreme cases shown in Fig. 15. Figs. 15b and 15c are generated by regularly and irregularly subdividing a local patch of Fig. 15a, respectively. Fig. 15b contains many small edges and Fig. 13c contains several near-degenerated triangles. In the experimental statistics (Table 2), both PSNR rates for vertex position and normal are high and the artifacts such as fold-over do not happen for these two extreme cases.

Table 3 shows the theoretical comparison of the three most related steganography approaches [27], [28], [29] and three watermarking approaches [12], [13], [16]. As expected, the steganography approaches provide more capacity but are less robust than the watermarking approaches. These four steganography approaches all belong to the blind detection schemes and hide data in the spatial domain. They can withstand similarity transformations and vertex reordering attacks (except [29]) because the vertex/face traverse orders are determined by PCA and a secret key. However, they cannot withstand attacks such as cropping, simplification, smoothing, and noise because either the vertex/face traverse orders are lost, therefore the embedded data is destroyed, or the vertices may potentially fall in incorrect regions after attacks. In other words, the embedded information cannot be recovered after suffering these malicious attacks. On the other hand, if we can find the correct vertex/face traverse orders (i.e., the receiver has the original model) after suffering the mesh simplification (assuming the simplification is done by a well-known QSLim algorithm with half-edge collapse operations [32]) or cropping attacks, we still have a chance to recover partial
information. If \( p \) percent vertices are cropped or simplified, we can extract \((100 - p)\) percent of the original embedded information correctly from attacked models (see Table 4 and Fig. 16). As a result, the proposed approach and these three related steganography approaches are not suitable for the applications of digital content protection and authentication. In terms of the similarity transformation attacks, our approach is less robust than the approaches [27], [28], [29] since our approach used more mantissa of the floating point representation. Any similarity transformation, such as translation or rotation, may shift the mantissa of the floating point representation of the coordinates, thereby potentially losing some of the bit positions that are used for embedding (i.e., approximately 1 percent to 2 percent loss of the embedded information in our experiments). However, high capacity is more important than the robustness in steganography techniques. In general, embedding payload in the spatial domain (3D space) can hide 3 bits per vertex such as the approaches presented in [28], [29], and our approach. To further increase capacity, we propose a multilayered embedding scheme that can embed data on many layers. The proposed approach offers a significant improvement in capacity compared to the previous 3D steganography methods, while the distortion is bounded.

Finally, we discuss our approach on the issue of security as follows: In all experiments, the distortion of the cover models suffered from our approach is very small, i.e., with PSNR rates > 81 dB for both vertex position and normal distortions. The visual difference between the stego and the cover models is imperceptible. Therefore, it is difficult to sense/determine whether the stego model contains any hidden messages. Our approach also allows a trade-off between the capacity and invisibility (see Fig. 11) by simply modifying the weighting coefficient \( \alpha \) in (10). If the
distortion factor is more important than the capacity factor, a large $\alpha$ can be selected. Furthermore, combining the secret key with our approach makes it harder to extract the hidden information. Therefore, we can produce better protection of the information if the stego object arouses suspicion.

6 CONCLUSIONS AND LIMITATIONS

We have introduced a novel multilayered information hiding scheme for 3D polygon models. We optimized the number of partitioning intervals to obtain a good balance between distortion and capacity. The experimental results showed that the proposed method can provide much higher capacity than previous approaches. Currently, the proposed approach has the following limitations that will be solved in the near future. Perfectly smooth (i.e., sphere) or extremely small-size models are not suitable for selection as cover models because the hidden data might be easily observed after even a very small modification by any embedding method. As in the previous work [27], [28], [29], utilizing PCA to determine the vertex traverse list may potentially hinder the robustness of the proposed method. Although the end vertices and initial triangle obtained from cover and stego models are identical in all cases in our experiments, we cannot guarantee that both are always exactly identical. One simple solution is to simply project vertices on the x, y, and z-axes. However, this approach cannot withstand similarity transformations. A better approach for determining vertex traverse list is required in the near future. Another limitation is that our approach cannot withstand certain malicious attacks such as smoothing, additional noise, nonuniform scaling, simplification, and vertices resampling. As a result, the proposed approach is not suitable for the applications of digital content protection and authentication.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Council (Contracts NSC-95-2221-E-006-193-MY2, NSC-96-2628-E-006-200-MY3, and NSC-96-2221-E-006-312-MY2), Taiwan, and in part by the Landmark Program of the NCKU Top University Project (Contracts B0008 and C0038). The authors would like to thank the anonymous reviewers for their insightful comments that help us improve this paper. The authors would also like to thank AIM@SHAPE Shape Repository, Stanford 3D Scanning Repository and Cyberware for the 3D polyhedral models used in this paper.

REFERENCES


Min-Wen Chao received the BS degree in mathematics and the MS degree from the National Cheng Kung University, Tainan, Taiwan, in 2003 and 2005, respectively. She is currently working toward the PhD degree in the Department of Computer Science and Information Engineering, National Cheng-Kung University. Her research interests include computer graphics and data hiding.

Chao-hung Lin was born in Koushung, Taiwan, in 1973. He received the BS degree in computer science/engineering from Fu-Jen University and the MS and PhD degrees in computer engineering from the National Cheng-Kung University, Tainan, Taiwan, in 1997, 1998, and 2004, respectively. He is currently an assistant professor in the Department of Geomatics, National Cheng-Kung University. His current research interests include computer graphics, image processing, visualization, and modeling.

Cheng-Wei Yu received the BS degree from the National Central University, Jhongli, Taiwan, in 2005 and the MS degree from the National Cheng-Kung University, Tainan, Taiwan, in 2007. He is currently with the Department of Computer Science and Information Engineering, National Cheng Kung University.

Tong-Yee Lee received the PhD degree in computer engineering from Washington State University, Pullman, in 1995. He is currently a full professor in the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan. He leads the Computer Graphics Group, Visual System Laboratory, National Cheng Kung University (http://graphics.csie.ncku.edu.tw/). His current research interests include computer graphics, nonphotorealistic rendering, image-based rendering, visualization, virtual reality, surgical simulation, medical visualization and medical system, and distributed and collaborative virtual environments. He is an associate editor for the IEEE Transactions on Information Technology in Biomedicine from 2000 to 2010. He is also on the editorial advisory board of the journal Recent Patents on Engineering, an editor of the Journal on Information Science and Engineering, and a region editor of the Journal of Software Engineering. He served as a member of the international program committees of several conferences including IEEE Visualization, Pacific Graphics, the IEEE Pacific Visualization Symposium, IEEE Virtual Reality, the IEEE-EMBS International Conference on Information Technology and Applications in Biomedicine, the International Conference on Artificial Reality and Telexistence, and the International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision. He is a member of the IEEE and the ACM.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.