Journal of Ambient Intelligence and Humanized Computing (2019) 10:4947–4961
https://doi.org/10.1007/s12652-019-01179-4

ORIGINAL RESEARCH

A reversible and affine invariant 3D data hiding technique based on difference shifting and logistic map

Ashish Girdhar[1] · Vijay Kumar[1]

Received: 14 July 2018 / Accepted: 3 January 2019 / Published online: 12 January 2019

Abstract
Image steganography has evolved as an active area of research because of its advantage to hide secret information inside an ingenuous image invisibly. In the recent years, improvements have been done in hardware to process 3D image models which laid down the path for 3D image steganography. The proposed work presents a novel approach of reversible data hiding in 3D mesh models. Difference between the vertices is shifted to hide secret bits inside the vertices. Chaotic logistic map is also used in the proposed work to decide the coordinate taken up for embedding. The proposed blind steganography system withstands rotation, scaling and translation attacks. Novel mesh traversal algorithm is proposed to visit the mesh vertices. This algorithm gives a unique mesh traversal algorithm for every 3D mesh model. The performance of proposed approach is compared with three well-known image steganography approaches over five cover models. The experimental results reveal that the difference between stego model and cover model cannot be observed by human eye. The proposed approach provides better embedding capacity than the previous approach in the field of reversible data hiding in 3D cover models.

Keywords Image steganography · 3D models · Chaotic logistic map · Reversible data hiding

1 Introduction

Image steganography has piqued the interest of researchers all around the world since the 9/11 attack (Cheddad et al. 2010). Advancements in technology and availability of internet in every nook and corner of the world has made all sorts of information readily available to everyone in a few clicks. Thus, security of confidential information over internet has become indispensable. In image steganography, secret information is hidden inside an ingenuous image in such a way that it remains invisible to the eye of intruders over transmission medium. 2D image steganography laid the foundation of 3D image steganography. 3D image steganography techniques use a 3D image model as cover model and secret information is taken as a stream of binary bits. The embedding of secret bits in 3D image model is a major concern. Distortion resulting from embedding may go unnoticed for 2D cover images as it is still a 2D image lying on paper. However, 3D image models reveal the unevenness (or distortion) caused on its surface due to hiding of information inside them when they are printed.

The main advantage of 3D image model over 2D cover image is that it is able to carry more secret bits than 2D cover images because of availability of more points (or vertices) in 3D image models. However, transmission of 3D stego model is costlier than 2D stego image. When large amount of information is to be sent is huge, then 3D image steganography should be preferred over 2D image steganography.

3D image steganography can be applied in every field where 2D image steganography is used. Whether it is for defence organisations purposes or protection of copyrighted material on internet, 3D image steganography finds its use. 3D cover models are larger carrying vessels than 2D cover images and thus for carrying larger volumes of secret data, 3D image steganography is used. In the field of medicine, 3D image steganography finds an interesting application. 3D models of human organs have been prepared, e.g. lungs, heart, etc. (Girdhar and Kumar 2017). 3D steganography can be used in these cases to embed information such as patient history, drug-allergy, etc inside these models.

3D image model consists of vertices and faces. The faces are connected through vertices. 3D image steganography refers to the process of hiding confidential

✉ Ashish Girdhar
ashishgirdhar410@gmail.com

[1] Thapar Institute of Engineering and Technology, Patiala, India

information in the form of text, audio, and video (Girdhar and Kumar 2017) inside 3D image model in such a way that it remains invisible to human eye. Secret data can be considered as a stream of bits. 3D image model used for hiding secret data bits is known as cover model. Many steganographic algorithms have been proposed for generating stego-model. There are two main approaches for 3D image steganography namely spatial domain and frequency domain (Girdhar and Kumar 2017). Spatial domain-based approach is preferred over frequency domain-based approach because of extra cost incurred in latter for taking 3D image model in and out from frequency domain (Girdhar and Kumar 2017). In spatial domain, information is hidden by altering geometry (Thiyagarajan et al. 2013), topology (Amat et al. 2010) or representation (Cheng and Wang 2006) of 3D image model. 3D mesh model or point cloud models can be used for steganography. 3D mesh models are used for the proposed algorithm.

In this paper, a novel approach on reversible data hiding in 3D image steganography is presented. A novel mesh traversal algorithm is also proposed in order to traverse the mesh vertices. Difference shifting technique is proposed for embedding bits inside vertices of cover mesh model. A vertex pair is selected and its difference is modified in order to hide secret bits. Original cover model is received back after extraction of secret bits. In this approach, chaotic logistic map is used for 3D image steganography which randomly chooses a coordinate value from the three coordinate values of a vertex for embedding process. Distortion caused can be reduced by changing only one coordinate value instead of three coordinates. The proposed technique modifies one coordinate value instead of three. The coordinate value is modified according to the outcome of logistic map. This would encourage the researchers of this area to further use chaotic maps in their works. The proposed steganography system uses its own mesh traversal algorithm for traversing over the mesh to withstand vertex reordering attack.

The remaining of this paper is structured as follows. In Sect. 2, background and related work are discussed. The proposed approach is described in Sect. 3. Section 4 discuss simulation results of the proposed approach on 3D mesh models. Some discussion related to the proposed approach is presented in Sect. 5. The concluding remarks are drawn in Sect. 6.

2 Background

In this section, the preliminary concepts of 3D image models and work done in the field of 3D image steganography are discussed.

2.1 3D image models

3D image models are images which have depth in addition to length and breadth. These models can be represented as a point cloud, NURBS surface, polygon mesh model, etc. However, polygon mesh model has higher transfer rates than the others (Girdhar and Kumar 2017). Due to this advantage, most research works prefer polygon mesh model for steganography technique.

The mathematical representation of 3D mesh is described below. A point in a 3D mesh is called a vertex. An edge is formed when two vertices are joined together. A face or polygon is a closed set of edges. A mesh containing triangle faces is known as triangle mesh and similarly the one with quadrilateral faces is quad mesh. Figure 1 shows 3D mesh model of a horse.

2.2 Related work

Many approaches have been developed for hiding secret bits inside 3D mesh models as studied in Girdhar and Kumar (2017). The authors focus mainly on reversible data hiding techniques on 3D mesh models. Ni et al. (2006) developed histogram shifting approach in reversible 2D image steganography. Based on it, some techniques in reversible 3D steganography have been developed. Jhou et al. (2007) and Chuang et al. (2010) constructed histogram of distances of all vertices from the gravity centre of mesh model. Largest and smallest histogram bars were discovered as peak and zero points respectively. Thereafter, the secret bits were hidden inside peak points. Distances of vertices from peak point to zero points were changed according to the modified histogram. In order to withstand vertex reordering attack, Jhou et al. (2007)'s scheme used the distances as the embedding order. However, it fails to extract the secret information correctly when the stego model has been attacked by rotation, scaling, or translation. Chuang et al. (2010)'s method was able to withstand affine transformations. It

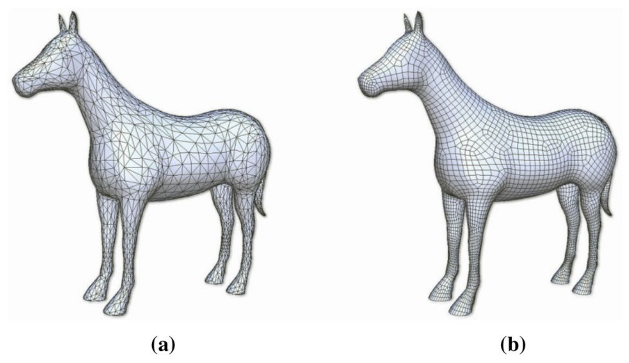


Fig. 1 **a** Triangle mesh, **b** quad mesh

used normalised distances of the points from centre for constructing histogram. Although this scheme is reversible with a blind extraction method. The main drawback is less embedding capacity. To enhance the embedding capacity, Huang and Tsai (2015) developed a reversible data hiding technique that uses distance of points with their neighbors for construction of histogram. This technique withstands vertex reordering attack because reference index of vertices is not taken for traversing over mesh. PCA was performed as a pre-processing step and then the initial triangle of mesh was chosen based on PCA axis. This made it withstand rotation, scaling and translation attack. However, the embedding capacity was still 0.51 bit per vertex. Ji et al. (2010) proposed reversible data hiding technique which is based on the concept of difference expansion idea (Tian 2003). They obtained distance ratio and used it in embedding process. Since the ratio of distances would remain indifferent towards changes in mesh due to rotation, scaling or translation. Thus, it could withstand these attacks. The embedding capacity was improved to 1.09 bit per vertex.

All the above-mentioned algorithms modify geometrical primitives of mesh. However, Tsai (2016) proposed a reversible steganography based on modifying topology of mesh. In this approach, the decimal equivalents of secret bits are generated. Using these values, mesh triangles are subdivided recursively till a threshold value is reached. This approach is able to withstand reordering attacks along with affine transformations. The embedding capacity is also improved from the above-mentioned approaches. Recently, a high embedding capacity algorithm was proposed by Li et al. (2017) which works by generation of a new truncation space for embedding of secret bits. The embedding capacity is more and extraction is blind. However, it is not reversible in nature.

As per the knowledge of authors, little work has been done in the field of 3D reversible data hiding techniques. Most of the existing techniques are based on histogram shifting scheme. These suffer from low embedding capacity. In 3D image steganography, the capability of 3D cover model is under-utilised. In this paper, the authors have tried to devise a novel reversible data hiding technique for 3D image steganography which could obtain a trade-off between capacity and distortion.

3 Proposed approach

A 3D mesh model has two arrays such as vertices and faces. It is observed from faces array that vertices are connected to each other. It helps in discovering neighbors from a particular vertex. The triangle faces of a mesh is shown in Fig. 2. Neighbors of vertex 1 are 2, 3 and 4. Neighbors of vertex 2 are 1 and 3. Similarly, 2 and 4 are neighbors for vertex 3 and vertex 4 has 1 and 3 as its neighbors. Such information about neighbors of vertices is required as a pre-requisite of the proposed approach.

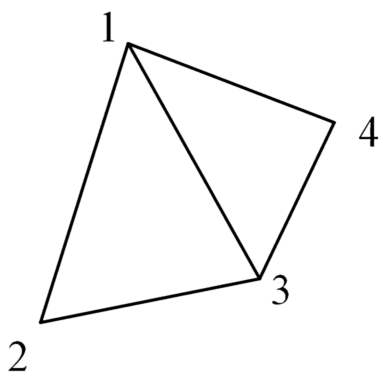


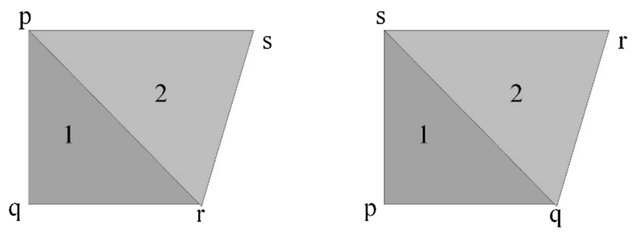
Fig. 2 Triangle faces of a mesh

3.1 Main idea

The main contribution of the proposed system is difference shifting scheme for reversible data hiding in 3D image steganography. In this method, a vertex pair is taken for hiding secret bits. From logistic map, one of the three coordinates such as x, y, z is selected. Value of vertex pair in selected coordinate axis is taken. Difference in this value is obtained which is modified in order to hide the secret bit. The new coordinate values are obtained using the modified distance. All the secret bits are hidden in a similar way. Vertices are referred as per the mesh traversal order generated by the proposed mesh traversal algorithm. Proposed steganography system hides secret bits in a reversible manner as the cover model is received back after extraction of secret bits from stego model.

3.2 Mesh traversal algorithm

The mesh traversal algorithm used in this approach, is based on shortest distance between neighbour vertices. It will generate the same referencing order every time over a particular mesh. Also, it should give only one traversing sequence without any ambiguity. Hiding bits inside mesh vertices should not alter the referencing order of vertices. It should remain exactly same during both embedding and extraction phases.



1	p	q	r
2	p	r	s

1	s	p	q
2	s	q	r

Fig. 3 Vertex reordering attack

Mesh traversal algorithm makes the stego-mesh resistant against vertex reordering attack. Vertex reordering of the mesh changes the reference index of vertices. Figure 3 shows an example of vertex reordering attack. In left mesh, face 1 consists of vertices *p, q*, and *r* while face 2 consists of vertices *p, r*, and *s*. In right mesh, face 1 has vertices *p, q*, and *s*; and face 2 has vertices *q, r*, and *s*. No change has been done in these two meshes except the reference indices of vertices of mesh. These meshes are vertex reordering attacks for each other. Thus, if these reference indices are taken for hiding bits, then the extraction procedure would fail.

Geometry and topology of the mesh remains intact but the reference indices are changed. Since only the vertex indices are changed; mesh looks similar as the previous one after attack. However, the retrieved information from stego mesh may be incorrect as the reference indices are different during embedding and extraction phases. Thus, it is a desirable characteristic of steganography algorithm to follow a mesh traversal algorithm while referring to vertices instead of indices.

The proposed mesh traversal algorithm is based on breadth first search. A source vertex is taken and all its neighbors are discovered and enqueued in the queue. The source vertex is a vertex nearest to the gravity centre of 3D mesh model. Then the distance to all the neighbors from source vertex is computed. The vertex nearest to the source vertex (i.e. minimum distance) is dequeued. Thereafter, the neighbors of dequeued vertex are discovered and enqueued into queue. This process goes on until all the vertices have been discovered and visited.

Even after vertex reordering attack, the proposed mesh traversal algorithm will visit the vertices in the same order. In Fig. 3, if the mesh traversal order is p, q, r, and s for mesh on left hand side, then it would be s, p, q, and r for the mesh on right hand side. Thus, the traversal of vertices of mesh is irrespective of their reference indices.

This proposed traversal algorithm gives same traversal order even if 3D model has been rotated, scaled or transformed. In case of rotation of 3D mesh model, distances between vertices would remain unaffected and thus same mesh traversal order would be generated. In case of uniform scaling, all the vertices would be scaled and hence no change in the traversal order would occur. In case of translation, 3D model in some other location would not alter the distances between the vertices and thus the traversal order remains intact. Thus, the traversal order is indifferent towards any changes in 3D model. It shows resistance against vertex reordering attacks. Algorithm 1 describes the proposed mesh traversal algorithm.

Algorithm 1 Mesh traversal algorithm

Input: 3D Mesh and source vertex

Output: Traversal order

```
1. Mark all vertices of Mesh as unvisited
2. Enqueue in Queue, Q the source vertex.
3. Mark source vertex as visited and put in traversal order
4. while (Q is not empty)
5.      v ← Q.dequeue( )          //Removing vertex whose neighbors will be visited
6.      Find all neighbours of v
7.      Calculate distances of all neighbours from v.
8.      for all neighbours w in ascending order of their distances from v
9.          if w is not visited
10.             Q.enqueue(w)
11.             mark w as visited and put in traversal order.
12.         end_if
13.     end_for
14. end_while
```

For better understanding of the proposed algorithm, Fig. 4 shows visiting queue in each step. Vertices being visited are shown in bold. Vertex 2 is nearer to vertex 1 than vertex 3. So, vertex 2 is enqueued prior to vertex 3 in visiting queue. The same process is repeated for all the vertices in the mesh.

3.3 Proposed steganography algorithm

The proposed steganography algorithm has been detailed out in three parts. In the first part, pre-processing step of the steganography algorithm is described. Pre-processing of mesh is required prior to both embedding and extraction procedures of steganography. In this step, information about 3D cover model is collected and traversal order is generated among other prerequisites of the proposed approach. In the second part of this section, embedding of secret bits has been detailed out. Extraction of secret bits from stego model has been discussed in the last part of this section.

3.3.1 Pre-processing

3D mesh model in which secret data is to be hidden, is first pre-processed. 3D mesh model has two arrays such as points and faces arrays. In case of triangle mesh, faces array has 3 columns, each row of which represents a face connecting three vertices. During pre-processing, a neighbor table is constructed from faces array. The neighbors of each vertex are searched and mentioned in a table. Afterwards, the proposed mesh traversal algorithm is applied on the mesh. A chaotic sequence is generated using chaotic logistic map which is given below.

$$a_{i+1} = \mu a_i(1 - a_i) \qquad (1)$$

where a_i is the chaotic sequence and μ is control parameter. For $3.569945 < \mu \le 4$, the system goes into chaotic state. Figure 5 shows the bifurcation diagram of logistic map. It is one of simplest chaotic maps. As per the best of knowledge of authors, the chaotic map is not used in 3D steganography

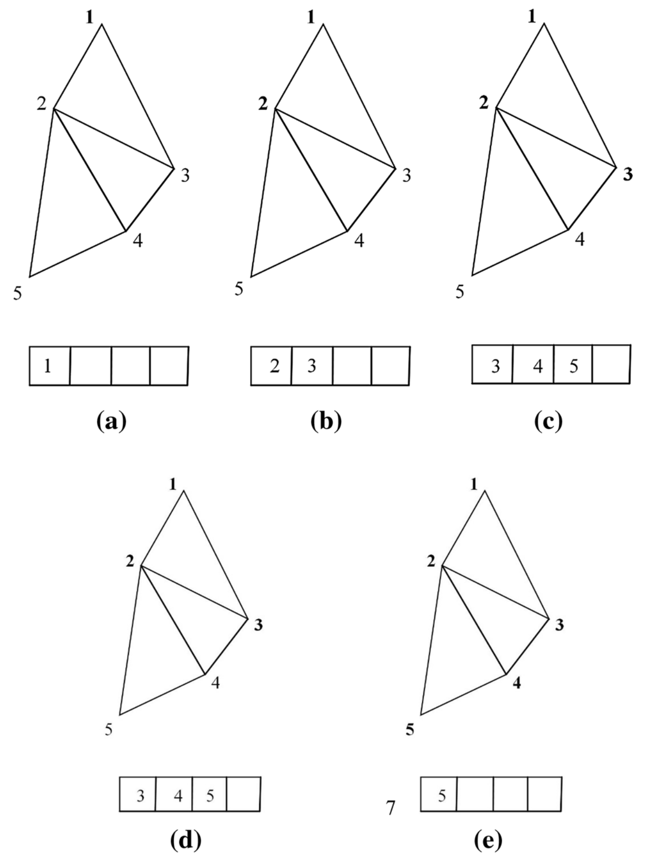


Fig. 4 Steps of mesh traversal algorithm from **a**–**e** along with queue

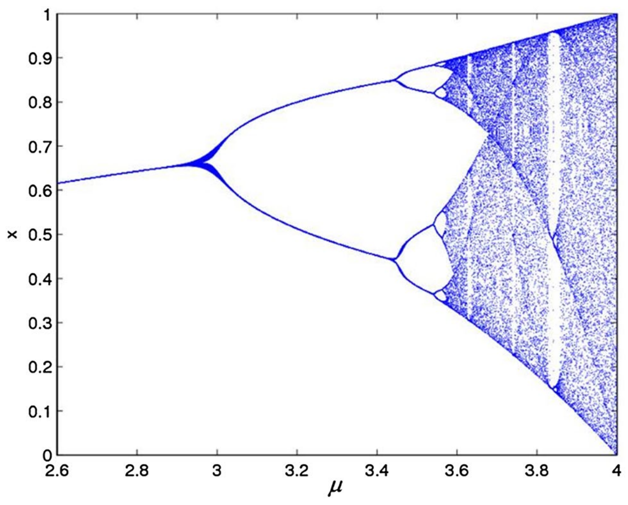


Fig. 5 Logistic map bifurcation diagram

system. Chaotic map is used in order to randomly hide data in one of the three coordinates of a 3D vertex, i.e., *x, y*, and *z* Andrecut (1998).

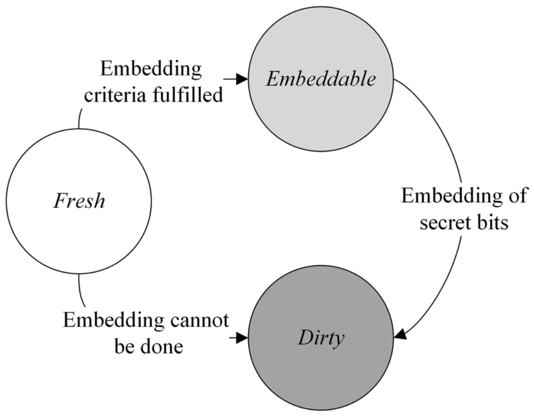


Fig. 6 Labels of vertices

Out of three coordinates of a vertex—x, y, and z; only one coordinate value is used for hiding secret bit. Values from logistic map are multiplied by 3 and ceil operation is applied in order to determine the coordinate for secret bit hiding among three coordinates.

There are three main advantages of using a chaotic map in 3D image steganography. First, secret bits are hidden randomly in the three coordinates. This makes the embedding pattern difficult to deduce for intruders. Second, secret bits are not hidden only in one-coordinate as done in Anish et al. (2017) where in x-coordinates only secret bits were hidden. Hiding in only one coordinate may cause more distortion to the cover model. Thus embedding of secret bits is done in all three coordinates. Third, given the exactly same initial control parameters, chaotic systems can be regenerated (Girdhar and Kumar 2018). The same chaotic sequences can be regenerated at the receiver end during extraction of secret bits that results in correct extraction of secret bits.

Chaotic map was first used by Matthews (1989) in image encryption field. Since then it is used by many image encryption approaches (Lan et al. 2018; Wang et al. 2016; Kumar et al. 2015; Safi and Maghari 2017; Sun et al. 2017; Chai et al. 2018). The proposed approach is the first 3D image steganography approach which uses a chaotic system. It may inspire other researchers of this area to use different chaotic maps in their approaches such as sine map, tent map, baker map, and/or their combinations. For the proposed approach, however the authors preferred chaotic logistic map keeping time and space complexity of chaotic systems in mind.

3.3.2 Embedding process

All vertices of the mesh model are labelled as—*fresh, dirty* and *embeddable*. All the unused vertices are labelled as *fresh*. From *fresh* vertices, an *embeddable* vertex(on which

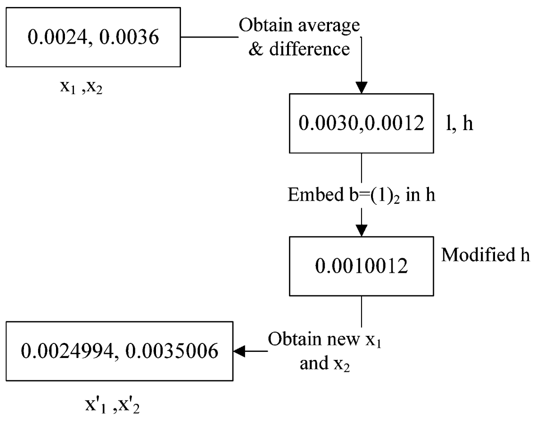


Fig. 7 Embedding based on difference shifting

embedding can be done) is chosen. It becomes *dirty* after embedding which implies it cannot be used anymore as shown in Fig. 6. A white vertex shows an unused vertex, while greyish-white implies that it can be used for embedding and a grey vertex shows that it cannot be used for embedding.

As soon as a vertex is discovered using the above-mentioned mesh traversal algorithm, it is labelled as *fresh* and checked if it is *embeddable*. If the *fresh* vertex does not satisfy embedding criteria it is marked as *dirty*, which implies it won't be looked up again as it is already *dirty*.

Number of neighbors associated with this *fresh* vertex is counted. The proposed steganography scheme is adaptive in nature in the sense that it distinguishes smooth and noisy surfaces. On a smooth surface, there will be less vertices or points than on a noisy surface. Embedding of secret bits is done inside the vertices present on noisy surface while the vertices of a smooth surface are kept undisturbed. It results in very less distortion caused by embedding of secret bits in 3D model. Since the number of vertices on a surface remains invariant to distortion less attacks (Girdhar and Kumar 2017). Hence, the proposed steganography algorithm withstands these attacks. Number of neighbors is a threshold parameter for embedding of secret bits. Once an *embeddable* vertex is found, its first *fresh* neighbor is taken and checked if this neighbor is *embeddable* too. If these two are *embeddable*, then they form the vertex pair on which embedding is done.

The proposed embedding algorithm is loosely based on difference between the coordinates of vertices. This method is based on the difference expansion by Tian (2003). Let the vertex pairs in which b-bit is to be embedded is (x_1, y_1, z_1) and (x_2, y_2, z_2). From logistic map values, the coordinate to be used for embedding is determined. Assuming that the secret bit is to be hidden inside x-coordinate of vertex pair. Average (l) and difference (h) between x_1 and x_2 can be computed as follows.

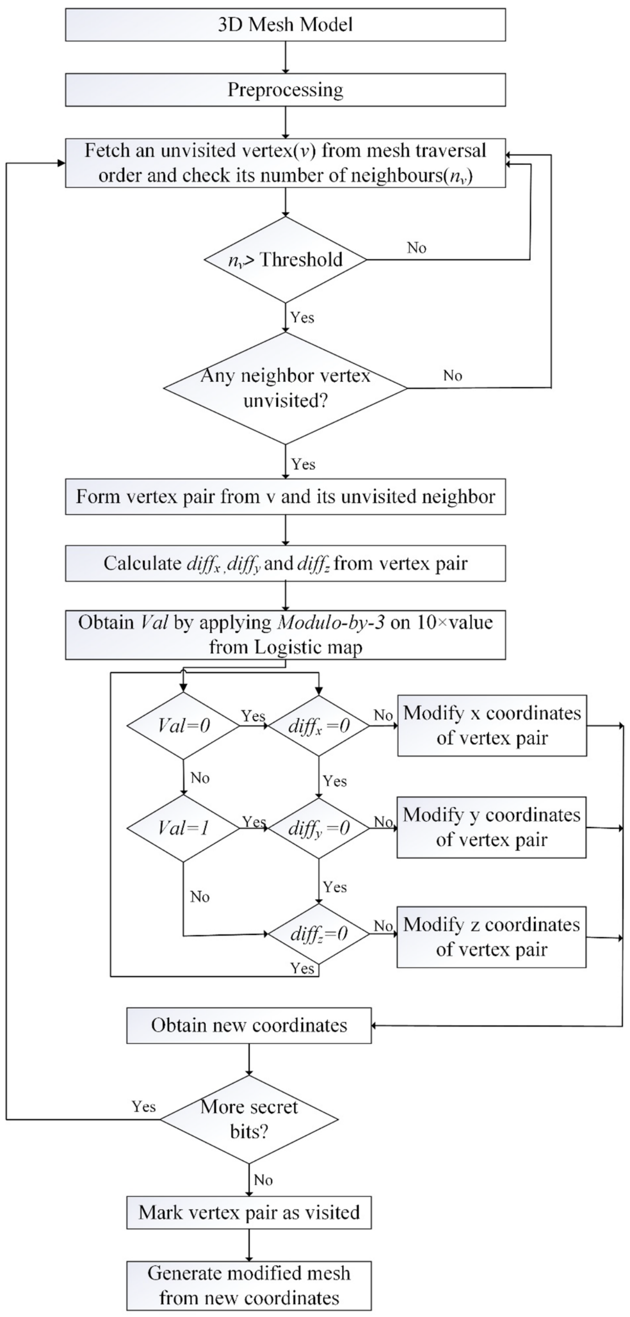

3D Mesh Model
Preprocessing
Fetch an unvisited vertex(v) from mesh traversal order and check its number of neighbours(n_v)
n_v> Threshold
No
Yes
Any neighbor vertex unvisited?
No
Yes
Form vertex pair from v and its unvisited neighbor
Calculate diff_x, diff_y and diff_z from vertex pair
Obtain Val by applying Modulo-by-3 on 10×value from Logistic map
Val=0
Yes
diff_x =0
No
Modify x coordinates of vertex pair
No
Yes
Val=1
Yes
diff_y =0
No
Modify y coordinates of vertex pair
No
Yes
diff_z=0
No
Modify z coordinates of vertex pair
Yes
Obtain new coordinates
Yes
More secret bits?
No
Mark vertex pair as visited
Generate modified mesh from new coordinates

Fig. 8 Proposed steganography system

$$l = (x_1 + x_2)/2 \tag{2}$$

$$h = x_1 - x_2 \tag{3}$$

Note that this difference is the absolute difference and hence it always a positive number.

Afterwards, a suitable value of ten is searched in order to hide secret bit in x_1 and x_2. This suitable power of ten is obtained by making use of x_1 and x_2. These values are multiplied by 10 repeatedly until the non-decimal parts of x_1 and x_2 are unequal.

For example, let us assume that$x_1 = 0.0024$ and $x_2 = 0.0036$. The difference between these points is 0.0012. Now, the values of x_1 and x_2 are multiplied by 10. The non-decimal parts of x_1 and x_2 are both equal to 0. So, they are multiplied with 10 again and still non-decimal part is equal. One more multiplication by 10 produces 2.4 and 3.6 in which non-decimal parts are unequal. Thus, the multiplication with 10^3 gives the desired result. From this particular case, it is observed the suitable power of ten is 3. Note that the actual coordinate values are not altered while performing this step.

The next step of proposed approach is secret bit hiding. The mathematical formulation is given below.

$$\grave{h} = \frac{h}{10^n} + \frac{\mathrm{b}}{10^n} \tag{4}$$

where $\grave{h}$ is the modified difference, b is decimal conversion of secret bit to be hidden and n is the suitable power of ten found above. Equation (4) is the difference shifting step in proposed 3D image steganography. Using the modified difference value, coordinate values are changed according to Eq. (5).

If $x_1 > x_2$

then $\grave{x}_1 = l + \grave{h}/2$; $\grave{x}_2 = l - \grave{h}/2$

Else

$$\grave{x}_1 = l - \grave{h}/2; \grave{x}_2 = l + \grave{h}/2 \tag{5}$$

where x_1 and x_2 are modified x-coordinate values of vertex pair.

Proceeding with the above-mentioned example of $x_1 = 0.0024$ and $x_2 = 0.0036$, average $l = 0.0030$, $h = 0.0012$ and assuming $b = 1$ the modified distance is calculated as $h = 0.0010012$. Using Eq. (5), the values of x_1 and x_2 are calculated as 0.0024994 and0.0035006 respectively. Figure 7 illustrates the embedding process based on difference shifting. It may be noted that no change is done in y and z coordinate value of vertex pair as the embedding is done only in x-coordinate of vertex pair. It is observed that the modified values are very close to the original values; thus, resulting in very less distortion. Similarly, when embedding is done in y and z coordinates of vertex pair then only those coordinate values are modified and other values are kept intact. This reduces distortion caused by embedding in all three coordinates as done in some previous reversible data hiding techniques for 3D models Chuang et al. (2010), Huang and Tsai (2015).

In the proposed approach, vertex pair is taken up for embedding secret bits. It is important to mark the *embeddable* vertex as *dirty* in order to differentiate it from the *fresh* vertices. After the modification of coordinate values, both vertices of vertex pair are marked as *dirty* so that no more modification is done on them. Thus, when selection of vertex pair for embedding, these *dirty* vertices are not selected.

Embedding capacity can be increased by a small change in embedding process. b is the decimal value of secret bit to be hidden. In place of taking 1 secret bit, 2 or 3 bits may be taken simultaneously and its decimal equivalent is then evaluated in order to hide secret information.

For instance, let the secret bit stream be ‘1011001001…’, instead of taking ‘1’ for hiding inside 3D model, ‘10’ or ‘101’ can be taken from bit stream. Thus, in place of hiding $1(=(1)_2)$ inside vertex coordinate, $2(=(10)_2)$ or $5(=(101)_2)$ will be hidden.

An interesting point to observe here is that for embedding, more than 3 bits cannot be taken at once from bit stream. This is because decimal equivalent of more than 3 bits may result in a 2-digit decimal equivalent and adding it to difference value will modify difference in such a way that the original difference cannot be retrieved.

For example, in the above-mentioned case, if $(1011)_2$ is taken, then $(11)_{10}$ is to be hidden. Let’s take original difference to be 0.5, shifting the difference would make it 0.05, Now addition of 0.05 and 0.11; makes new difference as 0.16. During the extraction process, 0.16 would wrongly give difference as 0.6 in place of 0.5! Thus, hiding more than 3 bits at a time inside a vertex pair destroys the reversible characteristic of proposed steganography scheme. The flowchart of proposed approach is shown in Fig. 8.

3.3.3 Extraction process

3.3.3.1 Response of stego-model to RST attacks The stego-model may be subjected to rotation, uniform scaling, and transformation on its way to the receiver. These attacks are distortion less attacks; i.e. no distortion is done to the mesh in these attacks. But these attacks are potent enough of destroying the secret bits embedded inside the mesh vertices. The proposed steganography algorithm withstands these attacks. Rotation causes rotation of mesh points around some point in space. In the proposed embedding process, angles between vertices are not manipulated or modified. Also, the rotation of mesh around some point would not alter distances among points, though the absolute difference between points will be altered. In case of translation of mesh to some other location, the distances would not be changed. In this case, the absolute differences would also remain same. Scaling would cause the distances among these points may increase or decrease. The average and difference needed for embedding, they would thereby have an impact of scaling of points in mesh. Thus, the secret information hidden may be incorrectly extracted from attacked stego-model subjected to rotation or scaling. So, in order to save the secret data being hampered by scaling of mesh, a simple solution is to undo the impact of scaling on the mesh.

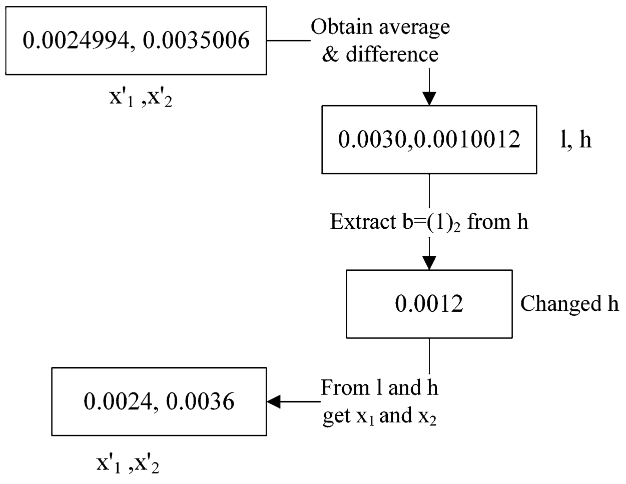


Fig. 9 Extraction process

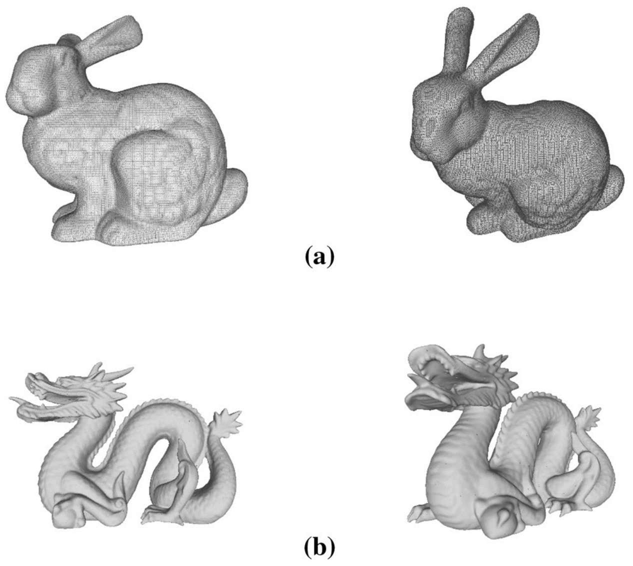


Fig. 10 Effect of rotation on bunny (**a**) and dragon (**b**)

Table 1 Description of cover 3D mesh models

Cover model	Number of vertices	Number of faces
Bunny	34,834	69,451
Dragon	100,250	202,520
Drill	1961	3855
Armadillo	172,974	345,944
Buddha	543,652	1,087,716

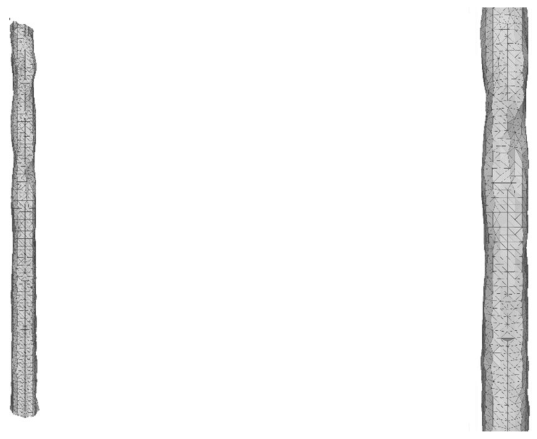


Fig. 11 Effect of scaling on drill

This is done by using 3D Helmert transformations for solving RST registration Gruen and Akca (2005).

$$\grave{P}_i = T(t_x, t_y, t_z) + SR(\omega, \varphi, \kappa)P_i \quad (6)$$

$$\begin{bmatrix} \grave{x} \\ \grave{y} \\ \grave{z} \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + SR(\omega, \varphi, \kappa) \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (7)$$

where $\grave{P}_i$is point from attacked stego model, P_i is corresponding point in stego mesh, $T(t_x, t_y, t_z)$ is translation vector, S is the scaling factor and

$$R(\omega, \varphi, \kappa) = \begin{bmatrix} cos\varphi cos\kappa & sin\omega sin\varphi cos\kappa - cos\omega sin\kappa & cos\omega sin\varphi cos\kappa + sin\omega sin\kappa \\ cos\varphi sin\kappa & sin\omega sin\varphi sin\kappa + sin\omega sin\kappa & cos\omega sin\varphi sin\kappa - sin\omega cos\kappa \\ -sin\varphi & sin\omega cos\varphi & cos\omega cos\varphi \end{bmatrix} \quad (8)$$

i.e. $R(\omega, \varphi, \kappa)$is rotation vector for rotating x, y and z axes using angles ω, φ and κ respectively. For solving these equations, three matched point pairs are required. Hence, these three matched point pairs are recorded in original stego model for matching with those in attacked mesh. Scaling effect on mesh model increases or decreases distances of all points among each other, so it would not affect the mesh traversing order. The points are compared to one another using the indices from mesh traversal algorithm. After solving the unknowns, the effect of RST on stego model is reversed in order to get back the stego model. This mesh model will be equal to the stego model sent from the sender side to receiver. Hence, it is given to the next stage of extraction of secret bits from it.

In extraction procedure, the same steps are followed as mentioned in Fig. 8. However, modifications are done in these steps to reverse the effect of hiding secret bits. The division is replaced by multiplication and addition is changed by subtraction. Logistic chaotic map is also used here at the receiver side with values of a_0 and μ from sender side. Multiplying values obtained from a_i chaotic sequence by 3 and then taking ceil value of it. It is revealed that the coordinate value of a particular vertex pair is hiding the secret bits. Then the embedding procedure is followed again but the operations are performed in reverse to obtain the original cover model back.

For instance, let the vertex pair to be processed is $(\grave{x}_1, \grave{y}_1, \grave{z}_1)$ and $(\grave{x}_2, \grave{y}_2, \grave{z}_2)$. From the logistic map, it is revealed that bit is hidden in x-coordinate. So only $\grave{x}_1$ and $\grave{x}_2$ are manipulated. The suitable power of ten, i.e., n is obtained again from $\grave{x}_1$ and $\grave{x}_2$.

Obtaining average

$$l = (\grave{x}_1 + \grave{x}_2)/2 \tag{9}$$

$$\grave{h} = \grave{x}_1 - \grave{x}_2 \tag{10}$$

$$h = \grave{h} \times 10^n - \lfloor \grave{h} \times 10^n \rfloor \tag{11}$$

$$b = \lfloor \grave{h} \times 10^n \rfloor - 1 \tag{12}$$

Now average and difference values are obtained, original x_1 and x_2 are obtained from Eq. (5).

In such a way, the secret bits from all points are retrieved and combined to get the secret information being hidden inside 3D stego model. The proposed steganography scheme has blind extraction method as it does not need the original cover model for extraction of secret bits. Also, it is reversible as it gives back the original cover model.

As illustrated in Fig. 9, taking the modified values of above-mentioned example, i.e. $\grave{x}_1 = 0.0024994$ and $\grave{x}_2 = 0.0035006$. Average calculated is 0.003, difference comes out to be0.0010012. The original difference can be restored back using Eq. (10) as0.0012 and the secret bit is revealed to be 1. Using the original difference and average of x_1 and x_2, they are given as $x_1 = 0.0024$ and $x_2 = 0.0036$.

Table 2 Embedding capacity of the proposed approach

Cover model	Threshold value	Size of payload attached (in bits)		
		Attaching 1-bit at a time	Attaching 2-bits at a time	Attaching 3-bits at a time
Bunny	4	17,237	34,474	51,711
Dragon	5	41,215	82,430	123,645
Drill	3	963	1926	2889
Armadillo	5	42,643	85,286	127,929
Buddha	6	63,429	126,858	190,287

4 Experimentation and results

The performance of the proposed steganography algorithm was tested on Intel i7-7500U processor running on Windows 10 operating system.

4.1 Experimentation set up and other prerequisites

The performance of 3D steganography is tested on different 3D mesh models obtained from Stanford graphics laboratory Computer graphics (2018) such as dragon, bunny, and drill. Table 1 depicts the mesh models to be used as cover models and their number of vertices and faces.

For embedding and extraction logistic chaotic map was generated with values $\mu = 3.7893$ and $a_i = 0.3333$. Stego-model was rotated, scaled, and translated using MeshLab (2018). The attacked stego-model is then corrected and extraction of secret bits was done. Effect of rotation can be observed on bunny and dragon in Fig. 10a, b respectively. In Fig. 11, drill was scaled uniformly. These are snapshots of the stego-models of these objects. Bunny and drill are wireframe snapshots of the respective 3D objects. Dragon snapshot is solid surface snapshot of dragon 3D model. Different snapshots are taken in order to observe the effect of steganography on surface of 3D models. As can be observed, these stego models withstand the effect of rotation, scaling and translation.

4.2 Performance metrics

In order to demonstrate the performance of proposed steganography approach, two performance metrics, namely capacity and distortion are used. Distortion should be as low as possible while capacity of the proposed approach should not be compromised. A trade-off between these parameters is always a daunting task for the algorithm designers. The proposed approach has been evaluated in terms of capacity and distortion.

4.2.1 Embedding capacity

Embedding capacity of a 3D steganography algorithm is the amount of secret data bits that can be hidden inside a vertex of 3D cover model. The proposed approach uses a single coordinate value of a vertex in order to hide 1 secret bit. Thus, the embedding capacity is 1 bit per vertex which is quite less than some previous works (Thiyagarajan et al. 2013; Cheng and Wang 2006; Tsai 2012).

Table 2 shows the embedding capacity of mesh models using the proposed approach. Embedding capacity can be however increased by a small change in embedding process as suggested above. Threshold values are taken as the

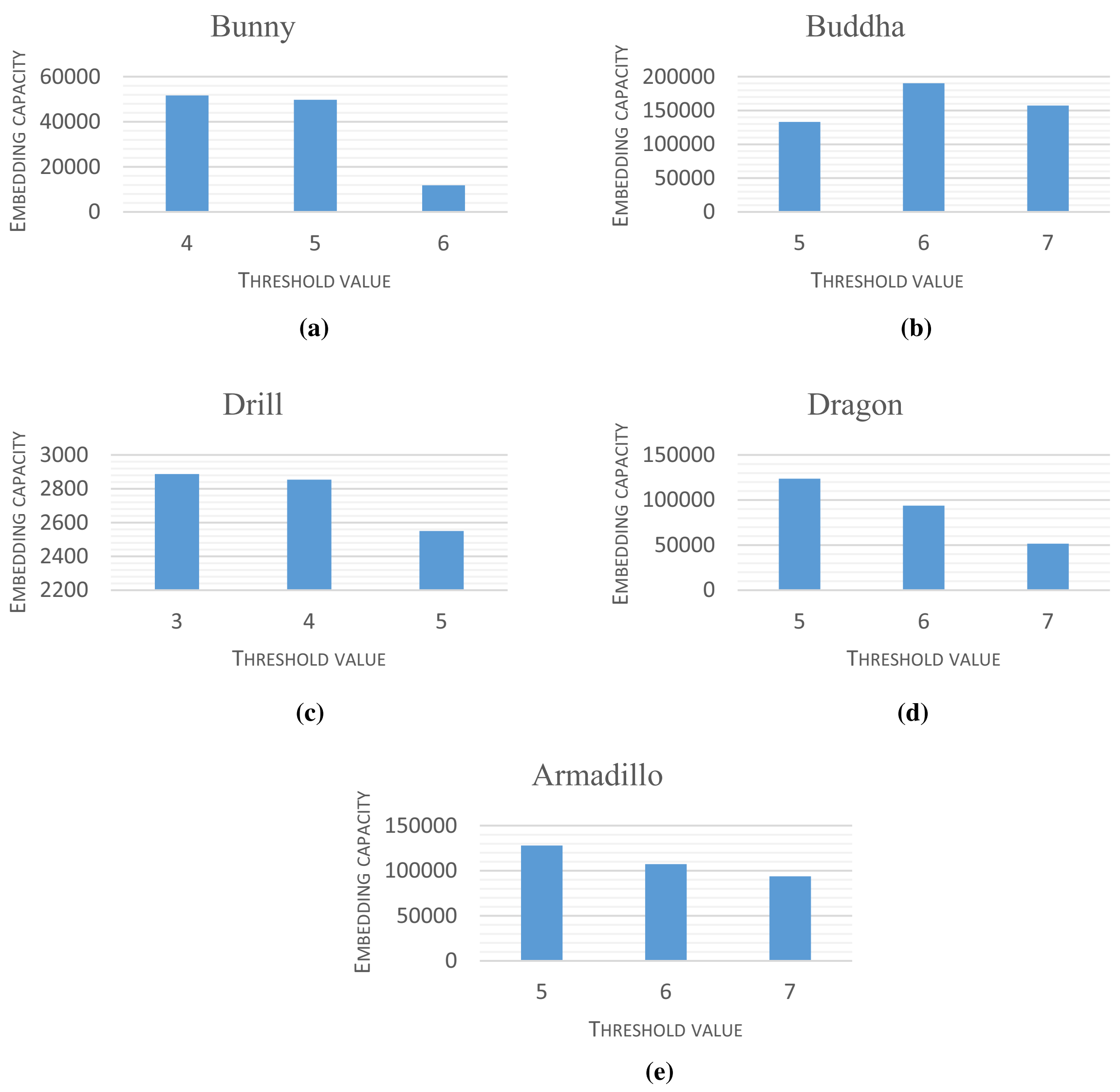


Fig. 12 Embedding capacity of **a** bunny, **b** buddha, **c** drill, **d** dragon and **e** armadillo cover models for different threshold values

Table 3 NHD values after hiding heavy payloads inside 3D meshes

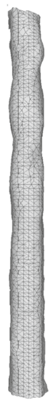
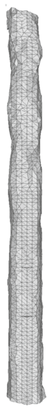
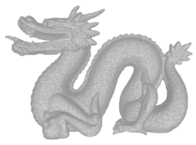
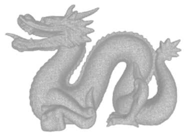
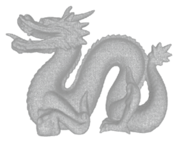
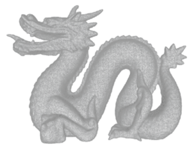
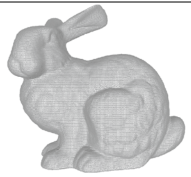
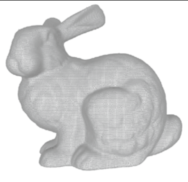
Cover model	Thresh-old value	Normalised Hausdorff distances (in 10^{-5})		
		Attaching 1-bit at a time	Attaching 2-bits at a time	Attaching 3-bits at a time
Bunny	4	0.60	0.75	0.98
Dragon	5	0.44	0.51	0.73
Drill	3	0.08	0.16	0.33
Armadillo	5	0.51	0.58	0.71
Buddha	6	0.48	0.52	0.67

average number of neighbors that the vertices of the mesh model have. When 1-bit of secret information is hidden at a time, less than half of the total number vertices carry the secret information. On the other hand, hiding 2 or 3 bits at a time increases the carrying capacity of the mesh models. It approximately doubles or triples the capacity, which is understandable. Altering threshold values would also have an impact on the carrying capacity of the mesh model. This can be seen in Fig. 12. These graphs are obtained by varying threshold values while embedding of secret bits. It is evident that in small mesh models like drill, a lower threshold value seems to be producing better embedding results than higher ones. This is reverse in case of larger mesh models like dragon. Simulation of the proposed approach has been done on other 3D models armadillo, buddha and bunny also.

Table 4 Comparison with previous techniques

Criteria	Algorithm			
	Chuang et al. (2010)	Huang and Tsai (2015)	Ji et al. (2010)	Proposed approach
Capacity (in bpv)	0.7	Upto 0.5	Upto 1.01	1–3
Adaptive	No	No	No	Yes
Domain	Spatial	Spatial	Spatial	Spatial
Blind extraction	Yes	Yes	Yes	Yes
Reversible	Yes	Yes	Yes	Yes
Robust against	RST	Reordering, RST	Some noise, RST	Reordering, RST

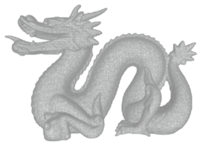
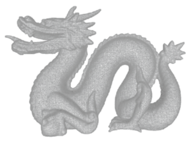
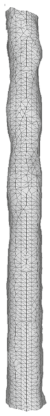
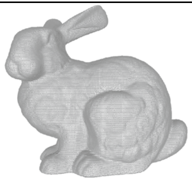
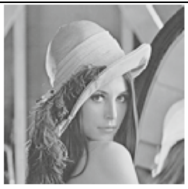
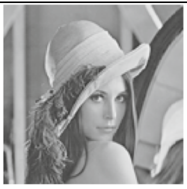
Table 5 Original and stego mesh models after hiding 1–3 bits of secret data

Model Name	**Original cover model**	**Stego model after hiding 1-bit**	**Stego model after hiding 2-bits**	**Stego model after hiding 3-bits**
Drill				
Drag —on				
Bunny				

4.2.2 Normalised Hausdorff distances (NHD)

Normalised Hausdorff distance (NHD) is a parameter of judging the similarity between stego model and cover model. This is very sensitive to any changes in two models. Normalised Hausdorff Distance is obtained by dividing Hausdorff Distance by diagonal length of the bounding box of mesh model. Values near 10^{-4} indicate visually acceptable distortion. From Table 3, it is evident that the values are acceptable in all the three cases. However, distortion on mesh may increases when the capacity goes high.

Table 6 Secret and extracted images by hiding 2 bits at-a-time in cover model

Model Name	Original cover model	Secret image	Stego model	Extracted image
Drag—on				
Drill				
Bunny				

4.3 Comparison with state-of-the-art techniques

In this section, the state-of-the-art reversible data hiding techniques are compared with the proposed approach. All the previously proposed schemes are not adaptive in nature. They simply embed bits inside vertices without bothering about the smoothness of a 3D surface. Thus, distortion caused to mesh model is bound to be more in case of these techniques. While all these techniques have blind extraction methods they also withstand the affine transformations. Table 4 lists out the comparison of the proposed approach with the previous works. Embedding capacity is increased in the proposed approach and adaptive embedding is done in this work.

Some of the differences observed in the above table are interesting to note. Works in Chuang et al. (2010) and Huang and Tsai (2015) are based histogram shifting mechanism of the 2D image steganography. Ji et al. (2010) used difference expansion idea on ratio expansion in 3D image steganography. Proposed approach modifies difference expansion technique to difference shifting. Embedding capacity of these works is still low. The proposed approach cannot withstand mesh simplification, smoothing or noise in mesh model because these will directly modify the vertex coordinates which carry secret data. However, Ji et al. (2010) withstands noise attack.

Table 5 shows 3D mesh models as cover and stego models. It is observed from the table that no visible changes can be seen with naked human eye. Thus, the proposed steganography system embeds secret data invisibly inside 3D mesh model. Wireframe model of the meshes has been included in order to be able to judge the similarity and/or dissimilarity between stego and cover mesh model. After extraction of secret bits from stego model, the original cover model is retrieved along with correct secret bits. Embedding capacity can be increased but at the cost of increasing distortion. 3D models having high number of faces and vertices (dragon) and low number of faces and vertices (drill) are used as example to evaluate the performance of the proposed approach on small as well as huge mesh models.

Table 6 shows secret images hidden inside respective 3D cover models along with the extracted images. As can be noted, secret and extracted images are absolutely similar to each other. Thus the extraction process successfully extracts the hidden secret image from stego-model without any loss of information bits. In Dragon and Bunny cover model, both lena and peppers of size 128×128 are hidden respectively. In Drill cover model, marbles image is hidden which is of size 64×64. Decimal-valued pixels of these images are first converted to their binary forms and then hidden inside 3D cover models.

5 Discussions and future scope

The proposed steganography algorithm is reversible and blind. Interesting outcomes of the work and future scope has been put below.

1. Mesh traversal algorithm proposed in the algorithm is based on breadth first search algorithm, visiting the nearest neighbor first. In case of rotation, scaling and translation attacks done on stego-mesh, proposed mesh traversal algorithm successfully gives the same traversal order over the mesh.
2. Embedding of secret bits inside vertices is done using a novel approach based on difference expansion technique of 2D image steganography. Difference expansion technique has been modified as difference shifting in this approach. Precision to store vertices of mesh becomes vital for correct working of proposed approach.
3. The proposed embedding algorithm takes help of a chaotic map for deciding which coordinate of the vertex should be used for embedding. Using a chaotic map hides secret bits in a pseudorandom way to make the pattern of embedding difficult to guess. In this work, logistic map has been used to serve the purpose. Other chaotic maps which give chaotic sequences can also be used in a similar manner. However, time complexity of chaotic map chosen should be small because lot of time goes in processing 3D cover model for embedding.
4. This approach is adaptive in nature as difference between smooth and noisy surfaces of the mesh has been taken into account while embedding of secret bits. A smooth surface will have less number of vertices than the noisy surface. Thus, the number of neighbors of a vertex on a smooth surface will be less than that of noisy surface. Hence, number of neighbors of a vertex is taken as the embedding criteria. Other embedding criteria distinguishing these surfaces can also be taken.

Lastly, the other reversible hiding techniques of 2D image steganography mentioned in Shi et al. (2016) can also be taken for 3D image steganography after modifying them as per use.

6 Conclusion

The proposed 3D image steganography system is reversible in nature and has blind extraction. It is loosely based upon the difference expansion idea of reversible data hiding in 2D images. It is also adaptive. This implies that smooth surfaces are left intact while the noisy surfaces are manipulated during embedding of secret bits. Mesh traversal algorithm is also proposed in this paper. The proposed steganography system is able to withstand vertex reordering attack. This algorithm gives a unique and only one traversal order of mesh irrespective of the times it has been run over a particular mesh. Also, the proposed approach uses chaotic map for the first time in 3D image steganography for hiding secret bits. Use of chaotic system is likely to motivate researchers of this area to use chaotic systems for their algorithms as well. Embedding capacity can still be improved in the proposed approach and hence remains the target to achieve for the authors.

References

Amat P, Puech W, Druon S, Pedeboy J (2010) Lossless 3D steganography based on MST and connectivity modification signal processing. Image commun 25(6):400–412. https://doi.org/10.1016/j.image.2010.05.002

Andrecut M (1998) Logistic map as a random number generator. Int J Mod Phys B 12(9):921–930. https://doi.org/10.1142/S021797929800051X

Anish K, Arpita N, Nikhil H, Sumant K, Bhagya S, Desai S (2017) Intelligence system security based on 3-D image. In: Proceedings of the 5th international conference on frontiers in intelligent computing: theory and applications, advances in intelligent systems and computing, pp 159–167. https://doi.org/10.1007/978-981-10-3153-3_16

Chai X, Zheng X, Gan Z, Han D, Chen Y (2018) An image encryption algorithm based on chaotic system and compressive sensing. Signal Process 148:124–144. https://doi.org/10.1016/j.sigpro.2018.02.007

Cheddad A, Condell J, Curran K, McKevitt P (2010) Digital image steganography: survey and analysis of current methods. Signal Process 90:727–752. https://doi.org/10.1016/j.sigpro.2009.08.010

Cheng Y, Wang C (2006) A high-capacity steganographic approach for 3D polygonal meshes. Vis Comput 22(9–11):845–855. https://doi.org/10.1007/s00371-006-0069-4

Chuang H, Cheng C, Yen Z (2010) Reversible data hiding with affine invariance for 3D model. In: Proceedings of IET international conference on frontier computing, theory, technologies and applications, pp 77–81. https://doi.org/10.1049/cp.2010.0541

Computer graphics (2018) Computer graphics at Stanford University. http://graphics.stanford.edu/data/3Dscanrep/. Accessed 19 June 2018

Girdhar A, Kumar V (2017) Comprehensive survey of 3D image steganography techniques. IET Image Proc 12(1):1–10. https://doi.org/10.1049/iet-ipr.2017.0162

Girdhar A, Kumar V (2018) A RGB image encryption technique using Lorenz and Rossler chaotic system on DNA sequences. Multimed Tools Appl. https://doi.org/10.1007/s11042-018-5902-z

Gruen A, Akca D (2005) Least squares 3D surface and curve matching. ISPRS J Photogramm Remote Sens 59(3):151–174. https://doi.org/10.1016/j.isprsjprs.2005.02.006

Huang Y, Tsai Y (2015) A reversible data hiding scheme for 3D polygonal models based on histogram shifting with high embedding capacity. 3D Res. https://doi.org/10.1007/s13319-015-0051-x

Jhou C, Pan J, Chou D (2007) Reversible data hiding base on histogram shift for 3D vertex. In: Proceedings of third international conference on international information hiding and multimedia signal processing, vol 1, pp 365–370. https://doi.org/10.1109/IIH-MSP.2007.268

Ji H, Yang X, Zhang C, Gao X (2010) A new reversible watermarking of 3D models based on ratio expansion. In: 3rd international congress on image and signal processing, Yantai, China. https://doi.org/10.1109/CISP.2010.564762

Kumar M, Powduri P, Redd (2015) A RGB image encryption using diffusion process associated with chaotic map. J Inf Secur Appl 21:20–30

Lan R, He J, Wang S, Gu T, Luo X (2018) Integrated chaotic systems for image encryption. Signal Process 147:133–145. https://doi.org/10.1016/j.sigpro.2018.01.026

Li N, Hu J, Sun R, Wang S, Luo Z (2017) A high-capacity 3D steganography algorithm with adjustable distortion. IEEE Access 5:24457–24466. https://doi.org/10.1109/access.2017.2767072

Matthews R (1989) On the derivation of a chaotic encryption algorithm. Cryptologia 13(1):29–42. https://doi.org/10.1080/0161-118991863745

MeshLab (2018) MeshLab, http://www.meshlab.net/. Accessed 1 June 2018

Ni Z, Su W, Shi Y, Ansari N (2006) Reversible data hiding. IEEE Transa Circuits Syst Video Technol 16(3):354–362. https://doi.org/10.1109/tcsvt.2006.869964

Safi H, Maghari A (2017) Image encryption using double chaotic logistic map. In: International conference on promising electronic technologies (ICPET), Deir El-Balah, pp 66–70. https://doi.org/10.1109/ICPET.2017.18

Shi Y, Li X, Zhang X, Wu H, Ma B (2016) Reversible data hiding: advances in the past two decades. IEEE Access 4:3210–3237. https://doi.org/10.1109/access.2016.2573308

Sun J, Liao X, Chen X, Guo S (2017) Privacy-aware image encryption based on logistic map and data hiding. Int J Bifurc Chaos. https://doi.org/10.1142/S0218127417500730

Thiyagarajan P, Natarajan V, Venkatesan V, Anitha R, Aghila G (2013) Pattern based 3D image steganography. 3D Res 4(1):1–8. https://doi.org/10.1007/3DRes.01(2013)1

Tian J (2003) Reversible watermarking using a difference expansion. IEEE Trans Circuits Syst Video Technol 13(8):890–896. https://doi.org/10.1109/TCSVT.2003.815962

Tsai Y (2012) An adaptive steganographic algorithm for 3D polygonal models using vertex decimation. Multimed Tools Appl, 69(3):859–876. https://doi.org/10.1007/s11042-012-1135-8

Tsai Y (2016) A distortion-free data hiding scheme for triangular meshes based on recursive subdivision. Adv Multimed 2016:1–10. https://doi.org/10.1155/2016/4267419

Wang X, Zhao Y, Zhang H, Guo K (2016) A novel color image encryption scheme using alternate chaotic mapping structure. Opt Lasers Eng 82:79–86