applied sciences

MDPI

Article

Interpolating Spline Curve-Based Perceptual Encryption for 3D Printing Models

Giao N. Pham [1], Suk-Hwan Lee [2] and Ki-Ryong Kwon [1,*]

[1] Department of IT Convergence & Application Engineering, Pukyong National University, Busan 608-737, Korea; ngocgiaofet@gmail.com

[2] Department of Information Security, Tongmyong University, Busan 608-711, Korea; skylee@tu.ac.kr

* Correspondence: krkwon@pknu.ac.kr; Tel.: +82-051-629-6257

Received: 3 January 2018; Accepted: 1 February 2018; Published: 5 February 2018

Abstract: With the development of 3D printing technology, 3D printing has recently been applied to many areas of life including healthcare and the automotive industry. Due to the benefit of 3D printing, 3D printing models are often attacked by hackers and distributed without agreement from the original providers. Furthermore, certain special models and anti-weapon models in 3D printing must be protected against unauthorized users. Therefore, in order to prevent attacks and illegal copying and to ensure that all access is authorized, 3D printing models should be encrypted before being transmitted and stored. A novel perceptual encryption algorithm for 3D printing models for secure storage and transmission is presented in this paper. A facet of 3D printing model is extracted to interpolate a spline curve of degree 2 in three-dimensional space that is determined by three control points, the curvature coefficients of degree 2, and an interpolating vector. Three control points, the curvature coefficients, and interpolating vector of the spline curve of degree 2 are encrypted by a secret key. The encrypted features of the spline curve are then used to obtain the encrypted 3D printing model by inverse interpolation and geometric distortion. The results of experiments and evaluations prove that the entire 3D triangle model is altered and deformed after the perceptual encryption process. The proposed algorithm is responsive to the various formats of 3D printing models. The results of the perceptual encryption process is superior to those of previous methods. The proposed algorithm also provides a better method and more security than previous methods.

Keywords: 3D printing security; 3D triangle mesh; spline curve interpolation; perceptual encryption; cryptography

1. Introduction

Three-dimensional (3D) printing is widely used in many areas of life [1,2]. Due to the fact that the benefits of 3D printing are enormous in many domains, 3D printing models are often attacked by pirates and distributed in commercial transactions without agreement from the original providers. The price of a 3D printer is cheap enough now that any individual user can buy a 3D printer, download 3D weapon models from the Internet, and print out real weapons such as knives and gun without any restriction. Moreover, certain special 3D printing models must be protected against unauthorized users. Consequently, security in the 3D printing industry is necessary.

Looking at 3D printing security techniques, watermarking methods have recently been proposed [3–5]. However, watermarking techniques do not alter the shape of 3D printing models. They only embed watermark data into 3D printing models, and anybody can view the shape of 3D printing models and design them again for printing. Therefore, watermarking is not suitable for secured storage and transmission; 3D printing models need to be encrypted. In addition, encrypting techniques must change the entire content of 3D printing models after the encryption process and be responsive to various model formats.

We propose an algorithm for perceptually encrypting 3D printing models. Facet data, the main component of 3D printing models, is used to interpolate a spline curve of degree 2 in three-dimensional space (3D space) that is determined by three control points, curvature coefficients, and an interpolating vector. The algorithm encrypts and distorts these features of the interpolating spline curve of degree 2 to obtain an encrypted 3D printing model. This paper is arranged as follows. We look into previous encryption techniques for 3D models and discuss 3D printing models related to the proposed method in Section 2. Section 3 presents the proposed algorithm in detail. In Section 4, we report visualization experiments, evaluations, and analysis of the proposed algorithm. Section 5 concludes.

2. Related Works

2.1. 3D Model Security

Watermarking techniques for 3D models have been researched. 3D model watermarking methods focus on the frequency domain and geospatial domain [6,7]. The key point of watermarking methods in the frequency domain is to embed watermark data into the spectrum coefficients of DFT, DWT, and DCT of a sequence of vertices or topologies, while the main concept of watermarking methods in the geospatial domain is based on modifying the value of vertices or geometric features such as length, area, or topology. Consequently, 3D model watermarking is not suitable for secured storage and transmission, as it fails to prevent attacks and illegal copying.

Some techniques that secretly share the content of 3D models and encrypt 3D CAD models have been proposed. Esam et al. [8] presented two methods that secretly transmit 3D models using Blakely, Thien, and Lin schemes. 3D models are separated into parts before they are shared, and users then reconstruct the 3D model from the shared parts of that model. However, this method is only for secured transmission. It is not an encryption method, and none of the 3D model is encrypted. Marc et al. [9] proposed a method by which 3D objects are encrypted based on geometry-preserving. This algorithm encrypts 3D objects by permuting the vertices of a given 3D object and distorting their shape while preserving its intrinsic geometrical properties. However, it does not alter the entire content of a 3D object and is not sensitive to the various formats of 3D objects. Moreover, the reconstruction cannot fully restore the original 3D object from the encrypted 3D object. Furthermore, the security of this method is very low. Cai et al. [10–12] proposed an encryption approach for 3D CAD models based on the geometric transformation of features of 3D CAD models. Here, a 3D CAD model is encrypted by randomization of the features of the 3D CAD model with an enhanced encryption transformation matrix. This method slightly changes the shape of 3D CAD models. Consequently, these methods do not achieve secured storage and transmission needed for 3D printing.

2.2. 3D Triangle Mesh-Based Encryption

A 3D triangle mesh has been used as an input for 3D printing [13,14]. A 3D triangle mesh is a set of facets. Each facet contains three vertices (a triangle) and a normal vector (see Figure 1). Each vertex is presented by three coordinates, x, y, and z. Thus, to encrypt a 3D triangle mesh, we only extract facets and encrypt all facets by the secret key. However, the normal vector of a facet only describes the direction of a facet and does not determine the shape of a 3D triangle mesh. Therefore, we only need to encrypt the triangles of 3D triangle mesh to generate the encrypted 3D triangle mesh.

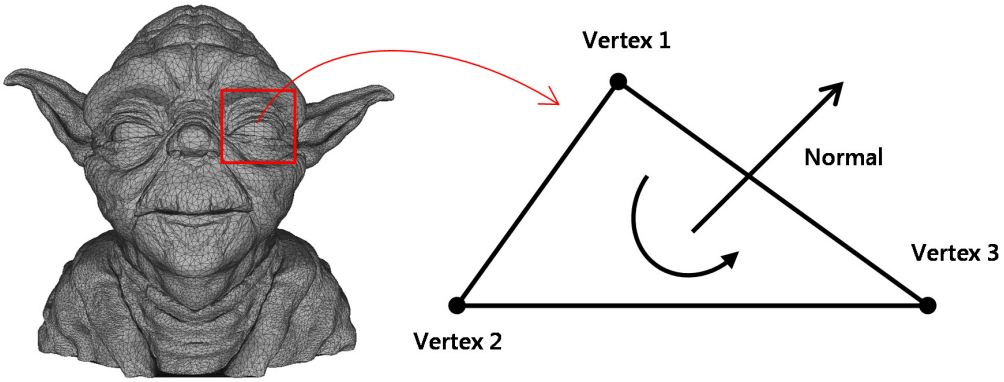


Figure 1. Structure of a 3D triangle mesh.

3. The Proposed Algorithm

3.1. Overview

Figure 2 describes the proposed algorithm. Each facet is extracted from a 3D triangle mesh. The index of each facet is then used with the key value **K** to generate an interpolating vector corresponding to that facet. The key value **K** is created from a user's key input by a key hashing function. The interpolating vector is then used to calculate the curvature coefficients of a spline curve of degree 2 in 3D space. These curvature coefficients of degree 2 are then used together with three vertices of facet to calculate three control points of the interpolated spline curve of degree 2 in 3D space. Next, the interpolating vector, curvature coefficients, and three control points of the spline curve are encrypted by the key value **K**. The encrypted curvature coefficients and the encrypted control points are used in an inverse interpolation process to compute three new vertices. These vertices will be distorted by the encrypted interpolating vector via the geometric distortion process in order to generate the encrypted facet. The encrypted facets are combined to obtain the encrypted 3D triangle mesh.

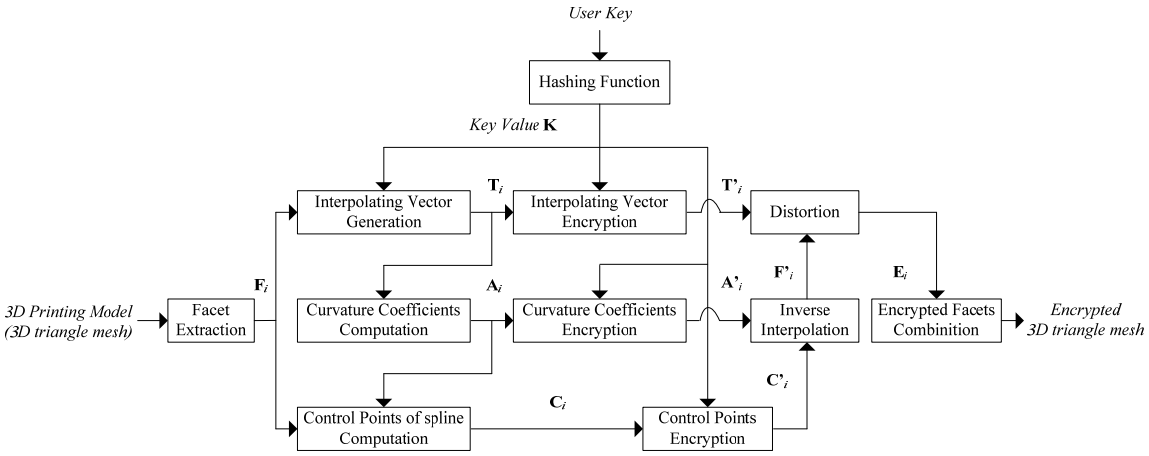


Figure 2. The proposed algorithm.

As mentioned above, a 3D triangle mesh contains a set of facets. Each facet includes three vertices. Each vertex is presented by x, y, and z coordinates. We assume that a 3D triangle mesh $\mathbf{M} = \{\mathbf{F}_i | i \in [1, |\mathbf{M}|\}$ with $|\mathbf{M}|$ the number of facets; $\mathbf{F}_i = \{v_{i,1}, v_{i,2}, v_{i,3}$ and $\mathbf{n}_i\}$ is the i^{th} facet with three vertices $\{v_{i,1}, v_{i,2}, v_{i,3}\}$ and the normal vector $\mathbf{n}_i(nx_i, ny_i, nz_i)$. Due to the fact that the normal

vector of a facet does not determine the shape of 3D triangle mesh, we briefly consider the facet $\mathbf{F}_i$ includes three vertices as Equation (1):

$$\begin{aligned} \mathbf{F}_i &= \{v_{i,1}, v_{i,2}, v_{i,3} | i \in [1, |\mathbf{M}|]\} \\ &= \{v_{i,j} | i \in [1, |\mathbf{M}|], j \in [1,3]\} \text{ with } v_{i,j} = (x_{i,j}, y_{i,j}, z_{i,j}). \end{aligned} \tag{1}$$

For brevity, main notations are defined as follows: $\mathbf{T}_i = \{t_{i,j} | j \in [1,3]\}$ is the interpolating vector corresponding to $\mathbf{F}_i$; $\mathbf{C}_i = \{c_{i,k} | k \in [1, 3]\}$ is three control points of the facet $\mathbf{F}_i$; $\mathbf{A}_i$ is the constructed matrix from the curvature coefficients of the interpolated spline curve corresponding to $\mathbf{F}_i$; $\mathbf{E}_i = \{e_{i,j} | i \in [1, |\mathbf{M}|], j \in [1,3]\}$ is the encrypted facet. Finally, $G_D(.)$ and $E_F(.)$ are the distortion function and the encryption function, respectively.

3.2. Features of the Interpolating Spline Curve in 3D Space

With each $\mathbf{F}_i$ we have three vertices $\{v_{i,j} | j \in [1,3]\}$. Each facet $\mathbf{F}_i$ is used to interpolate a spline curve of degree 2 in 3D space. This spline curve passes three vertices of the facet and is controlled by three control points (see Figure 3) [15,16]. However, these control points are determined by an interpolating vector and curvature coefficients. Thus, we could conclude that an interpolating spline curve of degree 2 in 3D space is determined by three control points, curvature coefficients, and an interpolating vector.

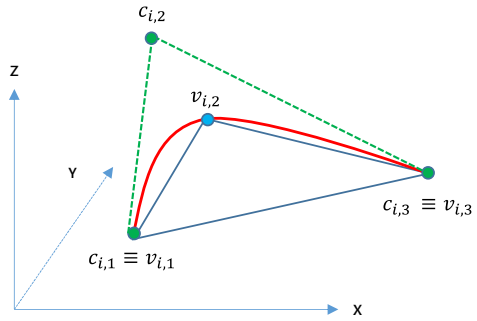


Figure 3. Example interpolating spline curve.

The interpolating vector $\mathbf{T}_i = \{t_{i,j} | j \in [1,3]\}$ is a set of numbers therein $t_{i,j} < t_{i,j+1}$. The interpolating vector $\mathbf{T}_i$ is then used to calculate the curvature coefficients. In order to respond to the condition $t_{i,j} < t_{i,j+1}$, we calculate $t_{i,j}$ by using indexes i, j, the value of $|\mathbf{M}|$ and $\mathbf{K}$ as described in Equation (2). The SHA-512 hashing function [17] is used to generate the key value $\mathbf{K}$ with a user's key input. Each key value has a length of 512 bits.

$$t_{i,j} = \frac{i \cdot (j+1)}{|\mathbf{M}|} \times \mathbf{K}. \tag{2}$$

With the three vertices of the facet and $\mathbf{T}_i$, we can compute the curvature coefficients and three control points of spline curve of degree 2. The relationship between three control points $\mathbf{C}_i = \{c_{i,k} | k \in [1, 3]\}$, $v_{i,j}$ and $t_{i,j}$ is defined as follows:

$$v_{i,j} = \sum_{k=1}^{3} B_{k,d}(t_{i,j}) \cdot c_{i,k} \tag{3}$$

where $B_{k,d}(t_{i,j})$ is considered the curvature coefficient of degree $d = 2$, corresponding to $t_{i,j}$ and the k^{th} control point, computed via the recursive function in Equations (4) and (5).

$$B_{k,d}(t_{i,j}) = \frac{t_{i,j} - t_{i,k}}{t_{i,k+d} - t_{i,k}} B_{k,d-1}(t_{i,j}) + \frac{t_{i,k+d+1} - t_{i,j}}{t_{i,k+d+1} - t_{i,k+1}} B_{k+1,d-1}(t_{i,j}) \tag{4}$$

$$B_{k,0}(t_{i,j}) = \begin{cases} 1 & if\ t_{i,k} \le t_{i,j} < t_{i,k+1} \\ 0 & otherwise \end{cases}. \tag{5}$$

From Equation (3), we can see that there are three curvature coefficients $\{B_{1,2}(t_{i,j}),\ B_{2,2}(t_{i,j}),\ B_{3,2}(t_{i,j})\}$ with each $t_{i,j}$. Therefore, we have 3×3 coefficients with the interpolating vector $\mathbf{T}_i$. These coefficients can be organized into a 3×3 matrix $\mathbf{A}_i$ in which the j^{th} row contains $B_{1,2}(t_{i,j})$, $B_{2,2}(t_{i,j})$, $B_{3,d}(t_{i,j})$:

$$\begin{aligned} \mathbf{A}_i &= \begin{bmatrix} B_{1,2}(t_{i,1}) & B_{2,2}(t_{i,1}) & B_{3,2}(t_{i,1}) \\ B_{1,2}(t_{i,2}) & B_{2,2}(t_{i,2}) & B_{3,2}(t_{i,2}) \\ B_{1,2}(t_{i,3}) & B_{2,2}(t_{i,3}) & B_{3,2}(t_{i,3}) \end{bmatrix} \\ &= \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \\ &= \{\, a_{j,k} \big| j,k \in [1,\ 3]\,\} \text{ with } a_{j,k} = B_{k,2}(t_{i,j}). \end{aligned} \tag{6}$$

Now, we apply three vertices of $\mathbf{F}_i$ and $t_{ij}|j \in [1,3]$ to Equation (3), the relationship between $\mathbf{F}_i$, $\mathbf{T}_i$, and $\mathbf{C}_i$ is described as follows:

$$\begin{aligned} \begin{bmatrix} v_{i,1} \\ v_{i,2} \\ v_{i,3} \end{bmatrix} &= \begin{bmatrix} B_{1,2}(t_{i,1}) & B_{2,2}(t_{i,1}) & B_{3,2}(t_{i,1}) \\ B_{1,2}(t_{i,2}) & B_{2,2}(t_{i,2}) & B_{3,2}(t_{i,2}) \\ B_{1,2}(t_{i,3}) & B_{2,2}(t_{i,3}) & B_{3,2}(t_{i,3}) \end{bmatrix} \cdot \begin{bmatrix} c_{i,1} \\ c_{i,2} \\ c_{i,3} \end{bmatrix} \\ &= \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \cdot \begin{bmatrix} c_{i,1} \\ c_{i,2} \\ c_{i,3} \end{bmatrix} \end{aligned} \tag{7}$$

$$\Leftrightarrow \mathbf{F}_i = \mathbf{A}_i \cdot \mathbf{C}_i$$

$$\begin{aligned} &\mathbf{C}_i = (\mathbf{A}_i)^{-1} \cdot \mathbf{F}_i \\ &\Leftrightarrow \begin{bmatrix} c_{i,1} \\ c_{i,2} \\ c_{i,3} \end{bmatrix} = \left(\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} v_{i,1} \\ v_{i,2} \\ v_{i,3} \end{bmatrix}. \end{aligned} \tag{8}$$

Finally, we can calculate three control points $\mathbf{C}_i$ from the curvature coefficients and three vertices of facet $\mathbf{F}_i$ as shown in Equation (8).

3.3. Features of Interpolating Spline Curve Encryption

After we compute the features of the interpolating spline curve in 3D space, we can encrypt them by the encryption function $E_F(.)$, which is the key value $\mathbf{K}$. We can use conventional encryption functions such as AES, DES, or XOR for the encryption function $E_F(.)$. Here, for simplicity, we encrypted features by the secret key value $\mathbf{K}$ as described in Equations (9)–(11), where $\mathbf{T}'_i$, $\mathbf{A}'_i$, and $\mathbf{C}'_i$ are the encrypted interpolating vector, the encrypted curvature coefficients, and the encrypted control points, respectively.

$$\begin{aligned} \mathbf{T}'_i &= E_F(\mathbf{K}, \mathbf{T}_i) \\ &= \{(\tfrac{\mathbf{K}}{|\mathbf{M}|} + j) \times t_{i,j} | j \in [1,\ 3]\} \\ &= \{\, t'_{i,j} \big| j \in [1,\ 3]\,\} \end{aligned} \tag{9}$$

$$\begin{aligned} \mathbf{A}'_i &= E_F(\mathbf{K}, \mathbf{A}_i) \\ &= \left\{ \tfrac{\mathbf{K}}{|\mathbf{M}|} \times a_{j,k} \big| j,k \in [1,\ 3] \right\} \\ &= \{\, a'_{j,k} \big| j,k \in [1,\ 3]\,\} \end{aligned} \tag{10}$$

$$\begin{aligned}\mathbf{C}'_i &= E_F(\mathbf{K}, \mathbf{C}_i)\\ &= \left\{\left(\frac{\mathbf{K}}{|\mathbf{M}|} + j\right) \times c_{i,j} \middle| j \in [1,\ 3]\right\}\\ &= \left\{c'_{i,j} \middle| j \in [1,\ 3]\right\}.\end{aligned} \tag{11}$$

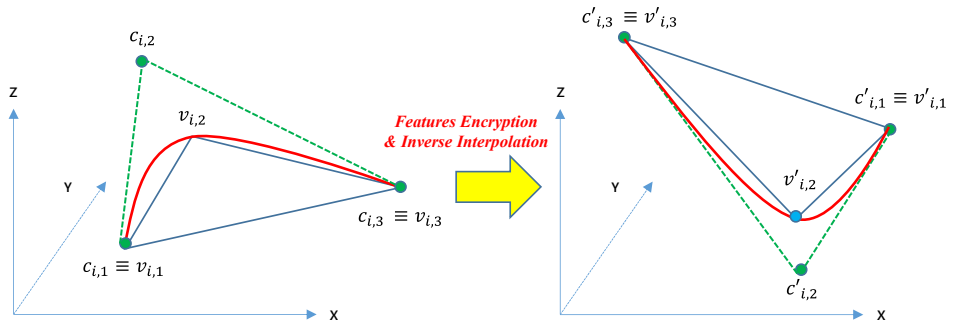
After the feature encryption process, the encrypted curvature coefficients $\mathbf{A}'_i$ and the encrypted control points $\mathbf{C}'_i$ are used in the inverse interpolation process in order to compute three new vertices as shown in Equation (12). Three new vertices are finally altered by the geometric distortion process in order to generate the encrypted facets $\mathbf{E}_i$. The geometric distortion process uses the encrypted interpolating vector $\mathbf{T}'_i$ as shown in Equation (13).

$$\begin{aligned}&\mathbf{F}'_i = \mathbf{A}'_i \cdot \mathbf{C}'_i\\ &\Leftrightarrow \begin{bmatrix} v'_{i,1} \\ v'_{i,2} \\ v'_{i,3} \end{bmatrix} = \begin{bmatrix} a'_{11} & a'_{12} & a'_{13} \\ a'_{21} & a'_{22} & a'_{23} \\ a'_{31} & a'_{32} & a'_{33} \end{bmatrix} \cdot \begin{bmatrix} c'_{i,1} \\ c'_{i,2} \\ c'_{i,3} \end{bmatrix}\end{aligned} \tag{12}$$

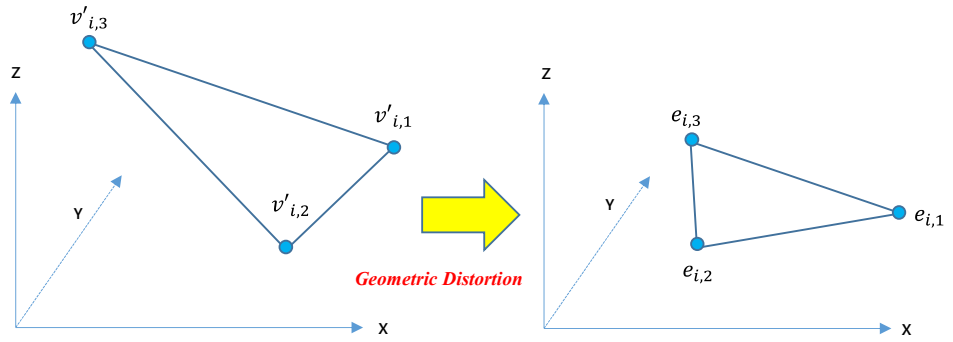
$$\begin{aligned}\mathbf{E}_i &= G_D(\mathbf{T}'_i, \mathbf{F}'_i)\\ &= \{(t'_{i,j} + 1) \times v'_{i,j} | j \in [1,\ 3]\}\\ &= \{e_{i,j} | j \in [1,\ 3]\} \text{ with } e_{i,j} = (ex_{i,j}, ey_{i,j}, ez_{i,j})\end{aligned} \tag{13}$$

$$\mathbf{E}_M = \{\mathbf{E}_i | i \in [1, |\mathbf{M}|]\}. \tag{14}$$

The encrypted 3D triangle mesh $\mathbf{E}_M$ is a set of the encrypted facets as shown in Equation (14). Figure 4 show the encryption process of a facet. Three control points are encrypted by the key value $\mathbf{K}$ and are then used to compute three new vertices $v'_{i,1}, v'_{i,2}$ and $v'_{i,3}$ by inverse interpolation (see Figure 4a). Three new vertices are formed into a new facet. Finally, this new facet is distorted by the geometric distortion to obtain the encrypted facets shown in Figure 4b.



(**a**)



(**b**)

Figure 4. Facet encryption: (**a**) features encryption and inverse interpolation; (**b**) geometric distortion.

3.4. Decryption Process

The decryption process is an inverse process of the encryption process. Each encrypted facet is also extracted from the encrypted 3D triangle mesh. The key value **K** and the index of each facet is used to generate the interpolating vector and the curvature coefficients as shown in Equation (2) and Equation (6) in Section 3.2. These features are then encrypted by the key value **K** that is used in Section 3.3. The encrypted interpolating vector and three vertices of the encrypted facet are used in the re-distortion process to compute three new vertices. These vertices are then used with the encrypted curvature coefficients to re-compute three encrypted control points. Three encrypted control points are then decrypted by the key value **K** in order to get three control points. Finally, three control points and the curvature coefficients are used to decrypt three vertices of the facet. The decrypted 3D triangle mesh is a set of decrypted facets.

The purpose of the proposed algorithm is to prevent illegal copying or illegal access from unauthorized users and attacks from hackers. Thus, if an authorized user or receiver would like to use the encrypted 3D printing models for 3D printing, they only need the secret key to decrypt and use them as 3D printing inputs. The decryption does not affect the printing process. To print a real object from a 3D triangle mesh, the 3D triangle mesh must be cut into a set of 2D slices and stored in G-code format. The content of the G-code format consists in paths that contain 2D coordinates for the CNC machine in a 3D printer. Consequently, the decryption process does not affect other aspects of the 3D printing process.

4. Experimental Results, Analysis, and Evaluation

We tested the proposed algorithm with 3D triangle meshes in Table 1. The format of the 3D triangle meshes was an STL file or a VRML file [13,14]. The information of the 3D triangle meshes is given in Table 1. In order to evaluate the proposed algorithm, we evaluated the perceptual encryption results of the 3D triangle meshes, the security of the encrypted 3D triangle meshes, and the computation time of the algorithm. Section 4.1 shows the perceptual encryption result of the 3D triangle meshes (visualization experiments). The security of the 3D triangle meshes and the computation time of the proposed algorithm are explained in Sections 4.2 and 4.3, respectively.

Table 1. Experimental results.

Name	# Facets	Entropy (dB)				Computation Time (ms)
		Proposed Method	Cai's Method	Marc's Method	Esam's Method	
Wheel Camping	544	4981	2754	2218	8.0	17.5
Knife	1176	12,032	6705	5430	8.0	29
Aperture Face	2026	22,292	12,485	10,137	8.0	46
Flower	2986	34,508	19,390	15,766	8.0	72
Gun	3961	47,378	26,681	21,714	8.0	90
Castle	7466	96,096	54,362	44,322	8.0	209
Batman	13,566	186,268	105,767	86,360	8.0	591
Rabbit	21,056	302,443	172,161	140,704	8.0	1312
Moto-bike	22,034	317,932	181,022	147,959	8.0	1421
Car	32,426	485,937	277,229	226,767	8.0	2951
Yoda	49,844	777,860	44,4681	364,021	8.0	6763
Lion	79,162	1,288,203	737,923	604,535	8.0	17,486

4.1. Visualization Experiments

To evaluate the visualization experiments, we evaluated the perceptual encryption results of the 3D printing models. The aim of the proposed algorithm is to alter and distort the shape of 3D printing models. Experimental results are shown in Figure 5. The number of facets in each 3D triangle mesh is different. The encryption process changes the values of the vertices of facets. Thus, the shape

of the facets is also altered. After the encryption process, facets are distorted into big facets (see "Encrypted Wheel Camping") or small facets (see "Encrypted Car" and "Encrypted Knife") or changed location, positioned disorderly (see "Encrypted Yoda, Encrypted Lion"). Thus, the shape of the 3D triangle meshes is altered. The content of the 3D triangle meshes is completely altered after the perceptual encryption process. Pirates or unauthorized users cannot extract or view the content of 3D triangle meshes.

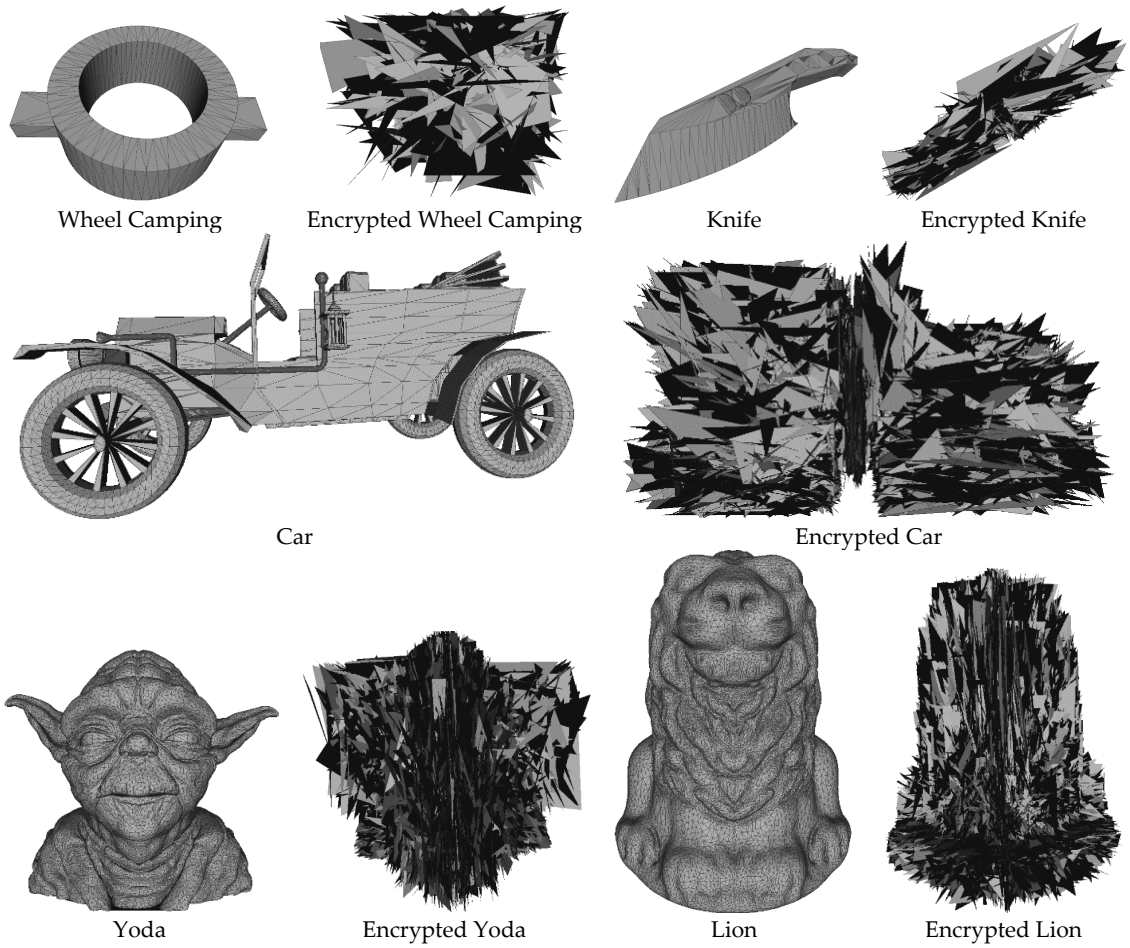


Figure 5. Experimental results with test models.

In Marc's method [9], he permuted the vertices of 3D meshes. According to his permutation, the shape of the 3D mesh changed, but he did not show the experimental results of his method. Moreover, the decryption of this method cannot fully restore the original 3D meshes from the encrypted 3D objects. In Cai's method [12], the encrypted CAD model was slightly changed (see Figure 6a,c). Anybody can view the content of the encrypted CAD model. We compared Cai's model with our proposed algorithm. Results are shown in Figure 6b,d. No unauthorized user could view or steal it, but the entire model was altered. Thus, the perceptual results of our proposed method are superior to those of Cai's method.

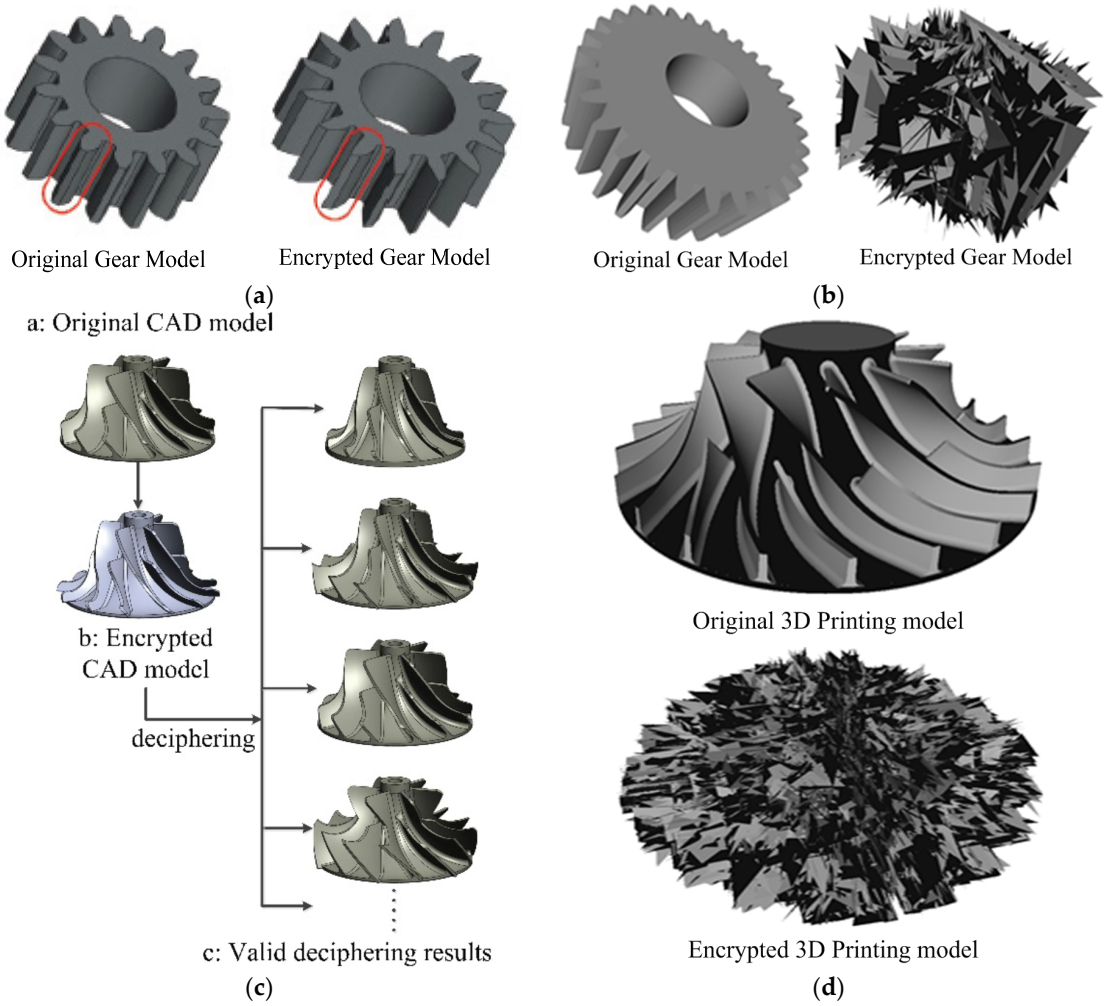


Figure 6. Perceptual encryption results of our proposed method and Cai's method. (**a**,**c**) Results of Cai's method; (**b**,**d**) results of the proposed method.

4.2. Security Evaluation

We analyzed the length of the secret key and the entropy of the encrypted 3D triangle mesh to evaluate the security of the proposed algorithm. If the secret key length is long, it is difficult to attack the encrypted 3D triangle mesh. If the entropy of the encrypted 3D triangle mesh is high, the security of the encrypted 3D triangle mesh will be high. In order to decrypt the encrypted 3D triangle mesh, a pirate must decrypt all of the encrypted facets of the 3D triangle mesh without the secret keys. In the proposed algorithm, the SHA-512 hashing function with a 512 bit salt is used to obtain random keys. Thus, if a user inputs a word with length L_k as his/her password, an attacker has to compute $L_k \times 2^{512}$ keys to access the encrypted 3D triangle mesh.

The entropy $H(x)$ of a discrete random variable x with a possible value $\{x_1, x_2, \ldots, x_n\}$ is defined as

$$H(x) = -\sum_{\tau=1}^{n} p(x_\tau) \cdot log_2 p(x_\tau) \tag{15}$$

where $p(x_\tau)$ is the probability density function of x on range $\{x_1, x_2, \ldots, x_n\}$. Based on the equations in Section 3, the entropy of the encrypted 3D triangle mesh is dependent on both the secret key $\mathbf{K}$, the interpolating vector $\mathbf{T}_i$, the matrix of curvature coefficients $\mathbf{A}_i$, three control points $\mathbf{C}_i$, and the number of facets $|\mathbf{M}|$. However, $\mathbf{K}$, $\mathbf{T}_i$, $\mathbf{A}_i$, $\mathbf{C}_i$, and $|\mathbf{M}|$ are random independent variables. Therefore,

the entropy of the encrypted 3D triangle mesh $H_{\mathbf{M}}$ is the sum of the entropies of variables $\mathbf{K}$, $\mathbf{T}_i$, $\mathbf{A}_i$, $\mathbf{C}_i$, and $|\mathbf{M}|$ and is determined by Equation (16):

$$\begin{aligned} H_{\mathbf{M}} &= H(\mathbf{K}) + H(\mathbf{T}_i) + H(\mathbf{A}_i) + H(\mathbf{C}_i) + H(|\mathbf{M}|) \\ &= |\mathbf{K}| \cdot \log_2 |\mathbf{K}|) + |\mathbf{T}_i| \cdot \log_2 |\mathbf{T}_i| + |\mathbf{A}_i| \cdot \log_2 |\mathbf{A}_i| + |\mathbf{C}_i| \cdot \log_2 |\mathbf{C}_i| + |\mathbf{M}| \cdot \log_2 |\mathbf{M}|). \end{aligned} \tag{16}$$

$|\mathbf{T}_i| = |\mathbf{C}_i| = 3$ and $|\mathbf{A}_i| = 9$. In summary, the total entropy $H_{\mathbf{M}}$ is dependent on $\mathbf{K}$ and the number of facets $|\mathbf{M}|$ in the 3D triangle mesh. If the secret key $\mathbf{K}$ is fixed, we can calculate the entropy of the encrypted 3D triangle mesh according to the number of facets $|\mathbf{M}|$ as shown in Table 1. The entropy of the encrypted 3D triangle mesh is formed from 4981 to 1.3×10^6 dB with $|\mathbf{M}| \in [544, 79162]$. From Equation (16) and Table 1, we can see that, if $|\mathbf{M}|$ is high, the entropy will be high.

Esam [8] divided the faces of a 3D model in N share hyper-planes before sharing. This mean a 3D model is divided into N parts before sharing. Thus, N is the secret key, and the entropy of Esam's method is always fixed. In Esam's method, $N = 4$, so the entropy of Esam's method is always 8 dB. Moreover, in the 3D space of the 3D model, the maximum number of hyper-planes is 360. Therefore, the maximum entropy of Esam's method is 3057 dB. Marc used the secret key $\mathbf{K}$ to encrypt and change the location of the vertices of the 3D mesh in OXYZ space. Clearly, Marc's method encrypted the vertices of the 3D mesh with a secret key $\mathbf{K}$. However, the number of vertices in a 3D mesh is always smaller than the number of facets. Thus, the entropy of this method is always lower than the proposed method. The entropy of Marc's method is formed from 2218 to 6.04×10^5 dB (see Table 1). Cai encrypted the features of the 3D CAD model with a random 3×3 matrix generated by a secret key. Thus, we can consider that Cai's method is encrypted 3D CAD models based on features and a random matrix by a secret key $\mathbf{K}$. Therefore, the entropy of this method is dependent on both the number of features and the 3×3 matrix. In the experimental results, around 50% of the facets are selected as the features of the 3D CAD model. Based on the test models in Table 1, the entropy of Cai's method is formed from 2754 to 7.38×10^5 dB. Figure 7 shows the entropy of the proposed method and the entropy of the previous methods according to the number of facets. The entropy of our method is always greater than the entropy of previous methods. Consequently, our method is superior, providing more security compared with previous methods.

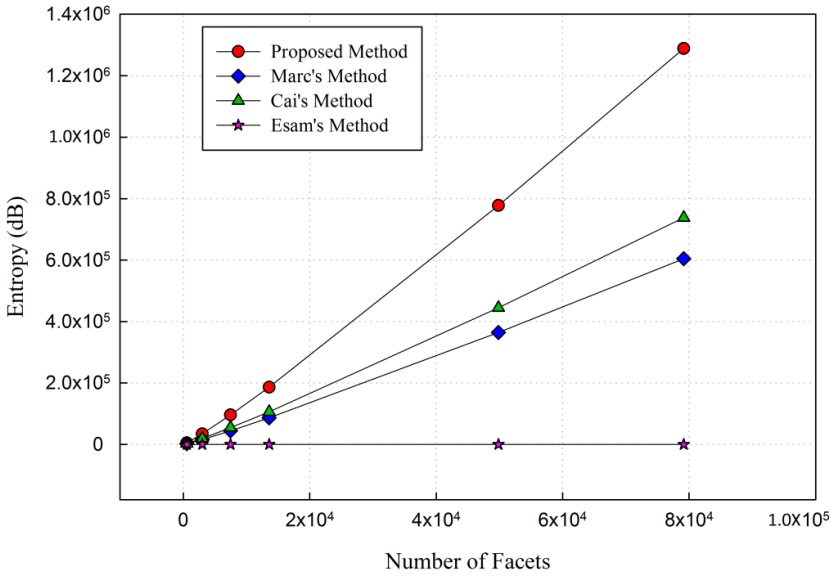


Figure 7. Entropy of the proposed method.

4.3. Computation Time and Analysis

We tested the proposed algorithm with a computer Intel Core i7-CPU 3.5 GHz; RAM-8 GigaBytes; Operating System: Windows 7 Microsoft–64 bits; Visual Studio 2013–C++. The computation time of the proposed algorithm was dependent on the number of facets in each model. With the test models in Table 1, computation time ranged from 17.5 to 17,486 ms with $|M| \in [544, 79162]$. Based on Table 1, we conclude that, if the number of facets is small, the computation time will be short.

Marc did not provide computation time. He only explained that his method will loop through all of the indices of the set of elements and will permute and swap at each iteration. As a result, the encryption time is expected to scale linearly with the complexity of the object. In addition, the decryption process requires swap operations to be performed in the reverse order than the one used for encryption. As a result, the decryption process is more "expensive" than encryption. We found that the encryption process of Marc's method was twice as long as our method and that the decryption process of Marc's method was three times as long. In our method, the encryption time and decryption time are similar. Moreover, the average computation time (including encryption and decryption times) of Marc's method is 1.25 times greater than the average computation time of the proposed method. In Cai's method, he only explained and analyzed factors affecting computation time. The computation time of Cai's method depends on how long it takes for the valid CAD model to check the time of feature encryption and the time of edge and facet encryption. He concluded that his method will meet users' requirements. We estimate that, with a computer has the same parameters, Cai's method must compute three parameters: valid model checking, feature encryption, and edges & facet encryption. We were able to estimate that the computation time of Cai's method is at least twice as long as our method. Therefore, we concluded that our method is faster than Cai's method as well. Figure 8 shows the computation time of the proposed algorithm and Cai's method with the test model in Table 1, according to the number of facets.

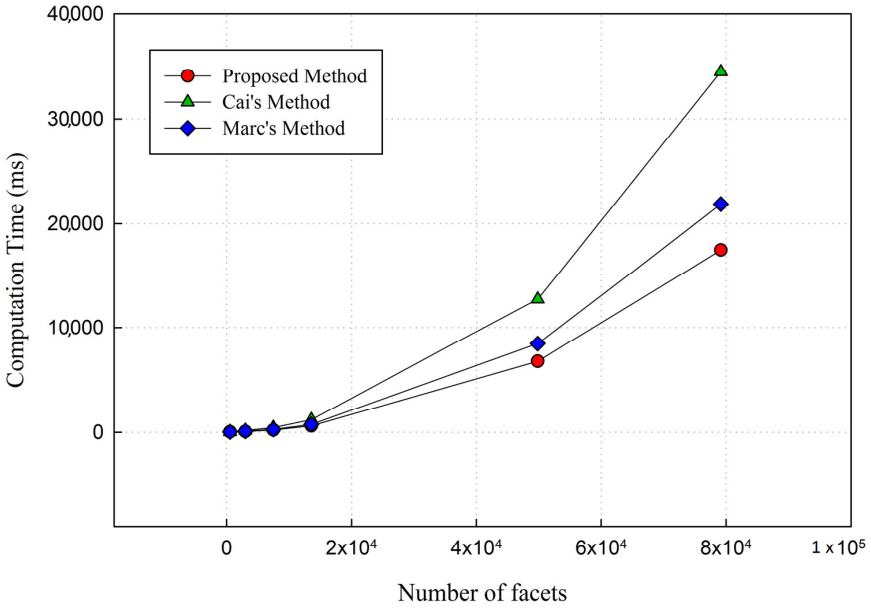


Figure 8. Computation time of the proposed method.

5. Conclusions

We present an algorithm for perceptually encrypting 3D printing models in this paper. The algorithm encrypts control points and curvature coefficients of the interpolating spline curve and geometric distortion. In light of the results of visualization experiments and evaluations, the proposed algorithm is shown to be effective and responsive to the various formats of 3D printing models.

The perceptual encryption results show that the proposed algorithm is more effective than previous methods; security is higher and computation takes less time. Therefore, the proposed algorithm provides more security than previously proposed methods. Our algorithm can be used for secured storage and transmission. However, the proposed algorithm does have limitations: computation is complex, and the algorithm is only useful for directly encrypting original 3D printing models. Some applications may require a combination between encryption and compression, and some may require real-time operation. We seek to improve our algorithm so that it can secure transmission systems and can be used in real-time applications and applied in compression domains. We hope to find other applications for our method as well.

Acknowledgments: This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2016R1D1A3B03931003 & NRF-2017R1A2B2012456), the MSIP (Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program (IITP-2017-2016-0-00318) supervised by the IITP (Institute for Information & communications Technology Promotion), and Brain Busan (BB21) by the Busan Metropolitan City.

Author Contributions: In this research activity, all of the authors joined and researched in the data analysis and preprocessing phases, the simulation, the results analysis and discussion, and the manuscript's preparation. All of the authors have approved the submitted manuscript. All the authors equally contributed to the writing of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. United States Government Accountability Office. *3D Printing Opportunities, Challenges, and Policy Implications of Additive Manufacturing*; DIANE Publishing Company: Collingdale, PA, USA, 2015.
2. 3D Systems Circle Rock Hill. *White Paper: How 3D Printing Works, The Vision, Innovation and Technologies behind Inkjet 3D Printing*; 333 Three D Systems Circle: Rock Hill, SC, USA, 2012.
3. Ho, J.U.; Kim, D.G.; Choi, S.H.; Lee, H.K. 3D Print-Scan Resilient Watermarking Using a Histogram-Based Circular Shift Coding Structure. In Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, Portland, OR, USA, 17–19 June 2015; pp. 115–121.
4. Yamazaki, S.; Satoshi, K.; Masaaki, M. Extracting Watermark from 3D Prints. In Proceedings of the 22nd International Conference on Pattern Recognition, Stockholm, Sweden, 24–28 August 2014; pp. 4576–4581.
5. Suzuki, M.; Piyarat, S.; Kazutake, U.; Hiroshi, U.; Takashima, Y. Copyright Protection for 3D Printing by Embedding Information inside Real Fabricated Objects. In Proceedings of the 10th International Conference on Computer Vision Theory and Applications, Berlin, Germany, 11–14 March 2015; pp. 180–185.
6. Tan, X.H. A 3D Model Asymmetric Watermarking Algorithm Based on Optimization Statistics. *J. Theor. Appl. Inf. Technol.* **2011**, *51*, 175–181. Available online: http://www.jatit.org/volumes/Vol51No2/2Vol51No2.pdf (accessed on 3 February 2018).
7. Ramya, P.; Nalini, D. A New Watermarking For 3D Models Based On Integral Invariants. *Int. J. Innov. Res. Comput. Commun. Eng.* **2013**, *1*, 190–192. Available online: https://www.ijircce.com/upload/2013/april/8_A%20New.pdf (accessed on 3 February 2018).
8. Esam, E.; Ben, A. Secret Sharing Approaches for 3D Object Encryption. *Expert Syst. Appl.* **2011**, *38*, 13906–13911. [CrossRef]
9. Marc, E.; Maetz, Y.; Gwenael, D. Geometry-preserving Encryption for 3D Meshes. In Proceedings of the Conference: Compression at Representation Signal Audio, Le Creusot, France, 28–29 November 2013; pp. 7–12.
10. Cai, X.T.; He, F.Z.; Li, W.D.; Li, X.X.; Wu, Y.Q. Encryption Based Partial Sharing of CAD models. *Integr. Comput.-Aided Eng.* **2015**, *22*, 243–260. [CrossRef]
11. Cai, X.T.; Li, W.D.; He, F.Z.; Li, X.X. Customized Encryption of Computer Aided Design Models for Collaboration in Cloud Manufacturing Environment. *J. Manuf. Sci. Eng.* **2015**, *137*, 1–10. [CrossRef]
12. Cai, X.T.; He, F.Z.; Li, W.D.; Li, X.X.; Wu, Y.Q. Parametric and Adaptive Encryption of Feature-Based Computer-Aided Design Models for Cloud-based Collaboration. *Integr. Comput.-Aided Eng.* **2017**, *24*, 129–142. [CrossRef]

13. STL Format in 3D Printing. Available online: https://all3dp.com/what-is-stl-file-format-extension-3d-printing/ (accessed on 3 February 2018).
14. The Virtual Reality Modeling Language. Available online: http://www.cacr.caltech.edu/~slombey/asci/vrml/ (accessed on 3 February 2018).
15. Shene, C.K. Michigan Technological University. Introduction to Computing With Geometry Notes. Available online: http://pages.mtu.edu/~shene/COURSES/cs3621/NOTES/ (accessed on 3 February 2018).
16. Lyche, T.; Morken, K. *Spline Methods Draft*; Dept. Informatics Centre of Mathematics for Applications, University of Oslo: Oslo, Norway, 2008.
17. RSA Lab. *Password-Based Cryptography Standard*; RSA Lab.: Bedford, MA, USA, 2006.