Advances in Science and Technology Research Journal

*Advances in Science and Technology Research Journal*, 18(8), 351–365
https://doi.org/10.12913/22998624/194146
ISSN 2299-8624, License CC-BY 4.0

Received: 2024.08.23
Accepted: 2024.10.04
Published: 2024.11.01

# 3D Model Fragile Watermarking Scheme for Authenticity Verification

Grzegorz Kozieł[1*], Liudmyla Malomuzh[1]

[1] Faculty of Electrical Engineering and Computer Science, Lublin University of Technology, ul. Nadbystrzycka 38A, 20-618 Lublin, Poland

* Corresponding author's e-mail: g.koziel@pollub.pl

## ABSTRACT

With the development of new technologies, 3D models are becoming increasingly important. They are used to design new models, document cultural heritage and scan valuable artefacts or evidence. They are also used in medicine. For these reasons, they are vulnerable to forgery. Protection against forgery done by encrypting the model or signing it digitally may restrict access to the data or require additional files to store the signatures. A good way to confirm the originality of 3D models is fingerprinting. This technique involves attaching a fragile watermark directly to the watermarked data. In the paper, we propose a new fingerprinting method for 3D models. The method hides the fingerprint in the least significant digits of the coordinates of the selected vertices. The fingerprint is created by calculating the hash-based message authentication code (HMAC) from the model textures and all vertex coordinates except the digits intended to attach the fingerprint. These digits are processed using discrete wavelet transform (DWT). The HMAC is attached to the selected DWT coefficients. The inverse discrete wavelet transform is then performed to obtain the new values of the modified digits. The digits are put back into the 3D model coordinates and the model is reassembled. Verification of the model originality is done according to the used steganographic key and consists of comparing the HMAC value extracted from the fingerprinted model with the HMAC value calculated from it. The same values of both HMAC results indicate that the model has not been modified. The proposed method allows efficient model fingerprinting and detection of changes made to any part of the model. The included fingerprints are transparent – the peak signal-to-noise ratio (PSNR) of a fingerprinted model can reach 150dB and its structural similarity can be over 99.8%. This paper presents a novel, computationally efficient fragile watermarking scheme that is capable of detecting the smallest changes in any part of a 3D model. The presented solution can be used to confirm the originality of 3D models. In particular, it will work well for fingerprinting large models such as 3D scans of architectural objects.

**Keywords:** steganography, fingerprinting, DWT, 3D, HMAC, PSNR, lifting scheme, fragile watermark.

## INTRODUCTION

With the development of new technologies, 3D techniques are gaining significance. They are used in modelling, industrial production, 3D printing and animation. A technology finding increasing application is additive manufacturing. It involves manufacturing parts by 3D printing based on a digital model. It is important to note that the print can be made from a variety of materials. The most common materials are thermoplastics and resins[1, 2]. Metals and composites are often used in industry[3]. It is also possible to print from ceramic materials[4] or food products[5]. 3D models are also widely used to document valuable artefacts [6] and can serve as a way of verifying their completeness. They are also used to store medical examination data or 3D models coming from motion capture [7, 8]. In many cases, protecting the 3D model against falsification or any modification is crucial [9, 10]. It is also important to verify its authenticity. Such protection can be done by cryptographic techniques, but they can be inconvenient by limiting access to the 3D model data. The best way of authenticity and originality verification is their fingerprinting. This

is because the originality proof is attached directly to the digital data which eliminates the risk of losing the proof. Most of the methods are designed to protect digital models [11, 12]. Some of them are designed in such a way as to enable watermark to survive the conversion of a 3D model to an analogue form [13, 14]. This can be important in the case of verification of the originality of some products [15] and their traceability [16]. In this case, digital data stops being connected to the model but an appropriately created watermark is possible to read from the printed model [17].

Fingerprinting, alongside steganography or watermarking, is one of the data hiding techniques. Whereas in steganography a high steganographic capacity and transparency are required, in watermarking robustness and imperceptibility are the most important [18]. Fingerprinting needs to be fragile and introduce as little distortion as possible. A fingerprint, also called a fragile watermark, is proof of data's originality. Fingerprinting aims to mark a set of data in such a way that even its smallest modification can be detected [19]. Any data modification should destroy the fingerprint or be doubtless evidence that the data has been tampered with [20]. Until now, few fragile watermarking methods for 3D models have been invented. Moreover, they are not without shortcomings, such as significant interference caused by a large number of modified bits and low computational efficiency.

To solve such problems and ensure a simple, computationally efficient and less prone to distortion verification mechanism, a novel fragile watermarking scheme for 3D models is proposed. The main contributions of the paper include the verification of the whole 3D model authenticity – its geometry and texture, an originality verification mechanism based on HMAC and discrete wavelet transform, as well as an examination of the influence on the interference level of attaching additional data into 3D mesh vertex coordinates.

The number of modified bits and the interference level were significantly reduced in the proposed method. No visual distortion is introduced. The method can detect a single-bit change.

The rest of the paper is organised as follows. The remainder of the introduction presents a review of 3D model fingerprinting methods. The 3D models, used transformations and the proposed method are described in the section on materials and methods. Discussion and results examine the influence of the changes introduced on model distortions, a discussion of the results obtained, and a comparison with other methods. A summary of the argument can be found in the conclusions section.

## Related works

In the literature, many robust 3D model watermarking methods can be found [21, 22]. They are used for content copyrighting [23]. Various techniques of embedding are being used in them, all aimed at obtaining high robustness of the attached watermark [24]. Each of these techniques attaches data in the defined domain. Spread spectrum methods are used to disperse changes in the whole model [22, 23]. Low-resolution approximation [25] helps to improve robustness. Vertex space is used to simplify attachment [26], and spherical harmonic transform space makes the attached data more difficult to identify [27]. Vertex norms distribution histogram bins or sparse quantisation index modulation space [26] are other techniques for improving robustness.

In the case of fragile watermarking methods robustness is not necessary so the most often vertex space is used to embed the watermark [27, 28]. Authors are also using the space of a mantissa of vertex coordinates, a model of connection points, spherical coordinate space, hash transform space, watermark digest or Karhunen-Loeve transform space [20].

In [29] a method for 3D video fingerprinting is proposed. The method uses depth-image-based rendering to attach a fingerprint. The goal of the method is to make it easier to find duplicated videos on the Internet. Due to this, the method ensures the fingerprint's robustness to certain attacks, similar to the procedure in [30], which relies on dividing the model into submeshes. The fingerprint is a combination of mean distances between the model gravity centre and groups of vertices; thus it is possible to confirm model originality even if some changes are introduced. In [31] a watermark embedding is done by introducing slight changes to the geometry of a 3D model. The method allows for the detection of which vertices have changed their positions. No information is given about how small changes can be detected. To verify texture's authenticity additional techniques have to be incorporated.

Paper [32] presents a semi-fragile watermarking method that hides data in integral invariants by changing the position of selected model vertices along with their neighbours.

The methods presented in papers [33, 34] attach the watermark data by making small changes to the geometry of the 3D object. An image is used as the watermark. Verifying the originality of the image requires comparing the extracted watermark with its original. In [35] a stegokey can be used, but it is optional and used rather as a cryptographic key to encrypt the payload before embedding. Paper [36] proposes hiding the watermark by modifying the position of selected model vertices in such a way that their position relative to the neighbouring vertices allows encoding fingerprint data. Due to the fact of modification of a significant number of vertices (38%-45%), the authors had to take care of reducing the introduced distortion and implement a mechanism for selecting vertices for modification in such a way that moving one vertex does not change the information encoded in another one. In [37] the problem of the influence of introducing changes to previously attached information was solved by marking triangles storing watermark data. Marked triangles and their vertices cannot be modified. The authors of [38] present a numerically stable fingerprinting algorithm based on modifying the position of selected vertices by modifying the values of their coordinates. The algorithm allows the detection of the locations of modifications to the model and their type. In [39] a watermarking method dedicated to computer-aided plant design topology protection is presented. A watermark is embedded by modification of a mantissa of real-valued sphere-polar coordinates. Under the term mantissa, we understand a part of a floating point number expressed as $a*10^n$, where: $a$ is the mantissa, $n$ is the exponent.

In work [35], the watermark is attached to the spherical coordinates system using the quantisation index modulation technique. The proposed approach achieves a low level of introduced distortion and avoids causality and convergence problems. In [40] genetic algorithm usage was proposed to reduce the level of introduced interference.

Apart from the above discussion, an attention should be put on the computational complexity of watermarking algorithms. All of these are processing all ($N$) vertices of the 3D model. The above solutions use such transformations as:

- discrete cosine transform having computational complexity $O(N\log_2 N)$,
- discrete Fourier transform having complexity $O(N^2)$ or $O(N\log_2 N)$ in the case of fast Fourier transform,
- discrete wavelet transform with a computational complexity of $O(N)$,
- other operations with a computational complexity of $O(N)$.

In the proposed algorithm an integer lifting scheme is used. It is an efficient algorithm for calculating wavelet transform based on the Haar wavelet. Its computational complexity is $O(N)$. Because it processes only a defined number of chosen coordinates (no more than 1536) it needs a smaller number of calculations.

## MATERIALS AND METHODS

### 3D models

Six models of various types, sizes and structures were utilised in the study. This diversity not only reflects the real-world variety of 3D models encountered but also enables the exploration of unique challenges and opportunities presented by each model type. The diversity in the origin of 3D models brings variations in spatial and material properties, which can significantly impact the effectiveness of steganographic techniques. For example, the material properties in a CAD model, such as reflectivity and texture, differ greatly from those in a photogrammetry-based model. The 3D models used are presented in Table 1.

The standard format for saving 3D models is “gltf”, which is saved as a binary file “glb”. This file contains a 12-byte header, chunk 0 storing JSON declarations and one or more data chunks containing vertices, textures and buffer views. The data chunks also encode the 3D model mode used in the file. Three modes of model representation are available: points, lines and triangles. This paper focuses on 3D models represented as a point cloud.

### Steganography

Steganography allows a payload to be hidden inside digital data. It aims to provide mechanisms for secret communication. This is achieved by hiding the information in another medium in such a way that it is challenging to detect the changes that have been made. Hiding is done by slightly modifying the original data. Many steganographic methods are known, such as the method of least significant bits (LSB) and its modifications, methods hiding data in the wavelet

**Table 1.** 3D models used in the research

| Model | Parameters | Model | Parameters |
|---|---|---|---|
|  | name: gate<br>size [kB]: 25600<br>number of vertices: 1088313<br>generation technology: scan 3D |  | name: tree<br>size [kB]: 5500<br>number of vertices: 213 292<br>generation technology: computer design |
|  | name: sculpture<br>size [kB]: 4700<br>number of vertices: 86938<br>generation technology: photogrammetry |  | name: pinecone<br>size [kB]: 13400<br>number of vertices: 58057<br>generation technology: Scan 3D |
|  | name: brick<br>Size [kB]: 136400<br>number of vertices: 17300<br>generation technology: scan 3D |  | name: dinosaur<br>Size [kB]: 2400kB<br>Number of vertices: 8527<br>Generation technology: computer design |

transform, Fourier transform, cosine transform and other transforms, pixel value differencing and many others [18, 41]. LSB methods hide data in the spatial domain, changing the position of vertices or the colour of chosen pixels of the texture. Methods hiding in a transform domain are at first performing a transform on original data and next attaching data by modifications of transform coefficients. Inverse transform allows for obtaining the original format of data. Some methods are based on artificial intelligence solutions to reduce introduced interference level [42]. Steganographic methods are designed to enable data to be hidden in sound, images, video, 3D models, and others [19, 43]. This is because of the various properties of these media. For each medium, another method will be appropriate. Each data type has specific properties to mask the presence of hidden data. An example would be images where frequency changes or slight colour changes of individual pixels are difficult to detect. Such features as steganographic capacity, robustness, transparency and security should be taken into consideration [18]. In the case of watermarking, high robustness and transparency are important, while in secret communication it is transparency and capacity that are significant. For fingerprinting also called fragile watermarking it is crucial to obtain a high transparency and fragility of the attached watermark [37].

## Discrete wavelet transform

The wavelet transform is one type of data transformation. As a result of the transformation, a new representation of it is obtained called the transform of the data. Wavelets are widely used in the field of signal processing, data compression and analysis [44, 45]. They are beneficial in analysing transients and waveforms characterised by high variability [46]. The continuous wavelet transform was proposed by Marlet-Grossman. For one-dimensional signals represented as an $x(t)$ function, it has the form:

$$w(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\, \psi * \left(\frac{t-b}{a}\right) dt \qquad (1)$$

where: $\psi(t)$ is the basis wavelet, * denotes the coupling of the function, $a$ is the scale parameter of the basis wavelet ($a>0$) while $b$ is its offset, and $t$ denotes time.

For discrete data, it is not possible to compute the continuous wavelet transform. In this case, the Discrete Wavelet Transform (DWT) is used, which is the equivalent of the continuous transform and allows you to work with digital data. The DWT is determined according to the formula:

$$a_{m,n} = \sum x(t)\,\psi_{m,n}(t) \tag{2}$$

In the discrete wavelet transform process, the vector of data is analysed using a basis wavelet, which is moved along the vector to be analysed and its correlation with the corresponding part of the signal is determined, resulting in a transform coefficient. Once the whole signal has been analysed, its representation is obtained. It consists of two parts: approximations and details. The detail contains the high-frequency components of the signal while the approximation of the signal is its low-frequency representation. This approximation can again be subjected to a wavelet transform obtaining results at the next level of decomposition. The wavelet transform is an iterative multiresolution analysis of the signal [47].

At each level of decomposition, a different level of the analysed signal details is obtained. This is used, among other things, in the process of de-noising the signal or removing unwanted components from it. In the case of the present work, the authors used wavelet transform coefficients to hide additional data by modifying their values. By performing an inverse wavelet transform, the data was reconstructed and the introduced changes were dispersed throughout the data vector. The inverse discrete wavelet transform (IDWT) is calculated according to the formula:

$$X2k = \sum ln-2kAn + hn-2kDn \tag{3}$$

where: the term $l_{n-2k}A_n$ involves the low-pass $l$ filter applied to the approximation coefficients $A_n$, and $h_{n-2k}D_n$ involves the high-pass filter $h$ applied to the detail coefficients $D_n$ [47].

## Lifting scheme

The lifting scheme is a variation of the Haar DWT, which first implements a lazy wavelet transform and then calculates detail integer coefficients. The lazy wavelet transform doesn't perform any calculations on the signal data. It separates the even and odd samples of the signal. It outputs two signals: one containing all the even-indexed samples and another containing all the odd-indexed samples. The signal given to this wavelet is subsampled into odd and even elements, which is done in an initial splitting step. The lazy wavelet transform is followed by the predict and update steps, which approximate input data. The goal of the predict step is to foresee that the *odd* element will be equal to the even element. The *odd* element in a new iteration equals the difference between the predicted function of the *even* value of the element and the previous *odd* element.

$$odd_{j+1,i} = odd_{j,i} - P\left(even_{j,i}\right) \tag{4}$$

Even elements instead are defined as an average of previous even and odd elements. Knowing this, an *even* element with index $i$ in iteration $j+1$ can be calculated by the formula:

$$even_{j+1,i} = even_{j,i} + U\left(odd_{j,i}\right) \tag{5}$$

where: $U$ is an update function, usually a scaling one, used to preserve certain features of the signal [48].

In the proposed method the lifting scheme was used. This transform was chosen because the inverse transformation returns integer values. This allows a simple and computationally efficient substitution of digits from the original vertex coordinates with digits from the inverse transformation result. Another reason for choosing the lifting scheme is this transform's low computational complexity (O($N$)).

## Fingerprinting method

In the designed method, the fingerprint will be hidden in the coordinates of the selected vertices. Each vertex of the model is represented by three coordinates specifying its position in the Cartesian coordinate system. The coordinate is a real number written to seven decimal places. This accuracy is due to the standard of storage of 3D models, which is GL Transmission Format (glTF). Data hiding is done by modifying the values of selected digits of selected vertex coordinates. The vertices' coordinates are numbered consecutively. The ones to be modified are then selected. Their numbers are indicated by values generated by a pseudo-random number generator.

### *Selection of digits to store fingerprint*

Current 3D models used in modern applications such as additive manufacturing or object evidence are characterised by a large number of vertices. The sequence containing the fingerprint consists of 512 numbers, each stored as a two-digit number with a sign (see section Fingerprint attachment). This gives a total of 1536 characters to encode. Attaching a watermark therefore requires 1536 digits to be modified in the vertex coordinates. Since each vertex is described by three coordinates, it is sufficient to modify the coordinates of 512 vertices assuming that only one of the numbers of each coordinate is modified. At the cost of introducing slightly more distortion into the model, it is possible to reduce the number of vertices used to 86. The 6 digits of each coordinate of the selected vertices are then used.

The selection of digits to hide the fingerprint starts by determining the number of vertices in the model ($N$). If this is greater than 512, the coordinates to be modified are selected. The selection of coordinates depends on the steganographic key. In the proposed implementation of the algorithm, the coordinates of the vertices were numbered sequentially with integers in the range from 1 to $n$, where $n$=3*$N$. A pseudorandom number generator initialised with a seed stored in the steganographic key was used. With it, $n$ integers are drawn at random to determine the coordinate numbers used to hide the fingerprint. Integers are drawn from the range (0-$n$>. This causes a uniform distribution of watermark in the whole 3D model. The scheme of the algorithm of vertices coordinates to use choose is presented in Figure 1.

One digit in each of the selected coordinates is modified. The selection of the digit is also controlled by the steganographic key. Preference should be given to the least significant digits due to the lowest level of distortion introduced.

If the number of vertices is less than 512, the fingerprint is hidden by using more digits of the selected coordinates. For models with fewer than

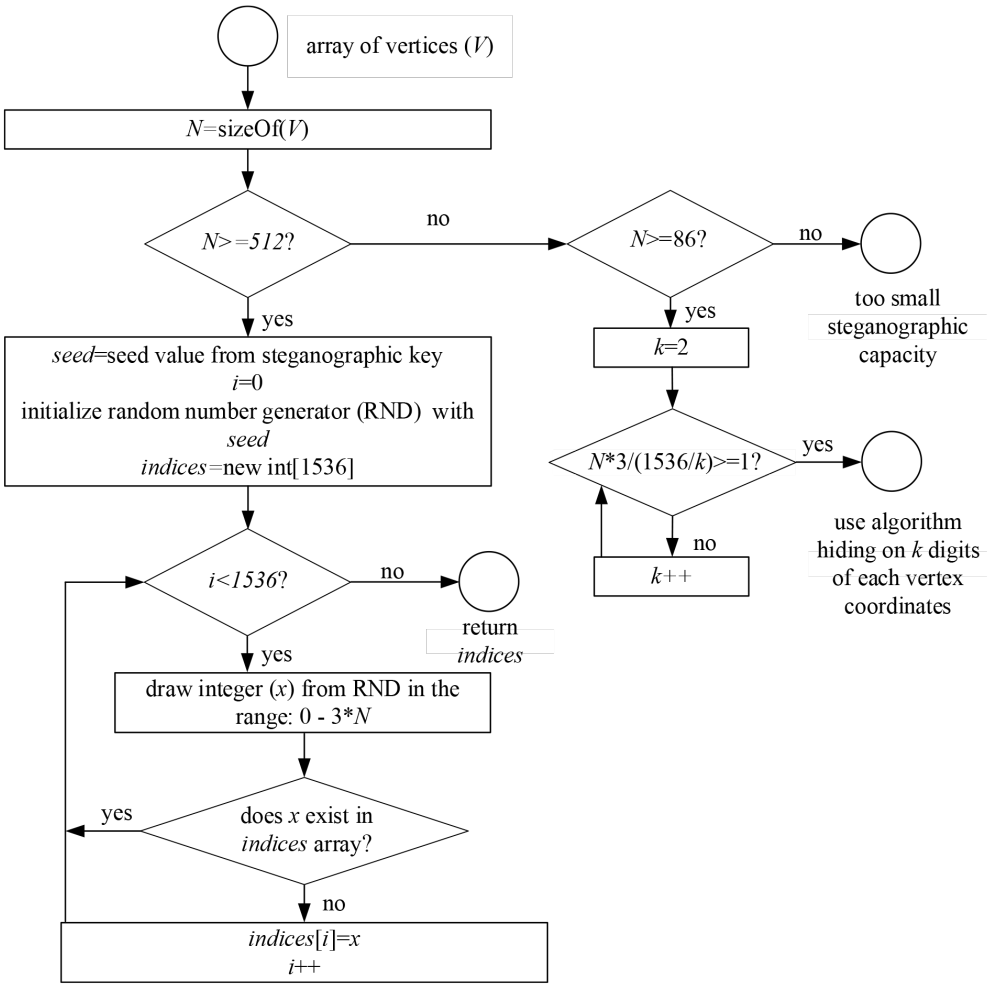


**Figure 1.** Scheme of the algorithm of coordinates choice

86 vertices, the proposed method does not work due to the insufficient steganographic capacity of the model. By steganographic capacity, we mean the size of data that can be hidden in a 3D model.

### *Fingerprint*

The fingerprint is the HMAC value calculated using the key stored in the steganographic key. The HMAC value is calculated from a dataset containing the 3D object's textures and vertex coordinates. HMAC can be calculated from any size data so there are no limits to the size of 3D model to fingerprint. Because the selected vertex coordinates will be modified when attaching the fingerprint, the calculation of their HMAC value must be preceded by an appropriate preparation. For this purpose, the digits selected to store the fingerprint are replaced with zeros. The set of vertex coordinates prepared in this way is appended to the dataset, from which the SHA512 HMAC value is then calculated. This operation results in a 512-bit long binary sequence, which is hidden in the previously selected digits of the vertex coordinates.

### *Fingerprint attachment*

The first 512 digits selected to attach the fingerprint are placed in a one-dimensional 512-element matrix (D). This matrix is transformed using a lifting scheme [48]. This operation results in a 512-element coefficient matrix. These coefficients are used to hide the previously calculated HMAC value. Hiding is implemented by subtracting or adding integers to the values of the coefficients. Modifying the coefficients of the lifting scheme by an integer value ensures that the inverse transform will generate integer values. The method of modifying the coefficients of the transform is controlled by the steganographic key. In its simplest form, an encoding of binary values can be used: an even value of the integer part of the transform coefficient corresponds to a binary zero and an odd value to a binary one.

The modified coefficient values are given to the input of the inverse lifting scheme. This transformation converts them to a set of 512 integers (S). These numbers are appended to the fingerprinted 3D model. However, due to the introduced modifications, they can take two-digit values and be negative. Therefore, it is not possible to directly replace the digits selected to hide the fingerprint with the obtained values. Each number from set *S* is encoded on three digits of the 3D model. The first one is used to encode the sign: an odd value of the number means a negative value, and an even value is a positive one. The second digit is replaced by the number of tens of the obtained number and the third by the number of unities. The use of the proposed coding makes it possible to include 512 obtained numbers by modifying the values of 1536 digits selected for fingerprint hiding. The scheme of the fingerprint attaching algorithm is presented in Figure 2.

### *Stegokey*

A steganographic key is a data set that controls the process of attaching data and extracting it. Without knowledge of the key, reading the hidden data is difficult and computationally inefficient [9]. In the proposed solution, the steganographic key includes:

- a seed to initialise a pseudo-random number generator allowing to determine the coordinates that will be used to hide the fingerprint,
- a secret key for calculating the HMAC value,
- the position of the digits to be modified,
- the algorithm for encoding the HMAC value in the values of the transform coefficients.

### *Fingerprint extraction*

To verify the originality of the model, it is necessary to read the hidden fingerprint. A steganographic key controls the reading process which starts from determining the set *R* of 1536 numbers that were used to hide the fingerprint. These numbers are identified using the mechanism used during fingerprinting. Next, they are passed to the decoding process. Its scheme is shown in Figure 3. As a result of decoding, an array *S* containing 512 numbers is obtained. Then it is processed using a lifting scheme. Values of the obtained coefficients are used to decode the HMAC value according to the algorithm shown in Figure 4.

### *3D model authenticity verification*

Verification of whether the model is original or has been altered is done by calculating the HMAC value of the 3D model according to the algorithm described in the fingerprint section and then comparing it with the HMAC value extracted from the fingerprint. If the two values are identical, this confirms that no changes have been made to the model since it was fingerprinted. An inconsistency in the compared HMAC values means that the model was subject to modifications after fingerprinting.

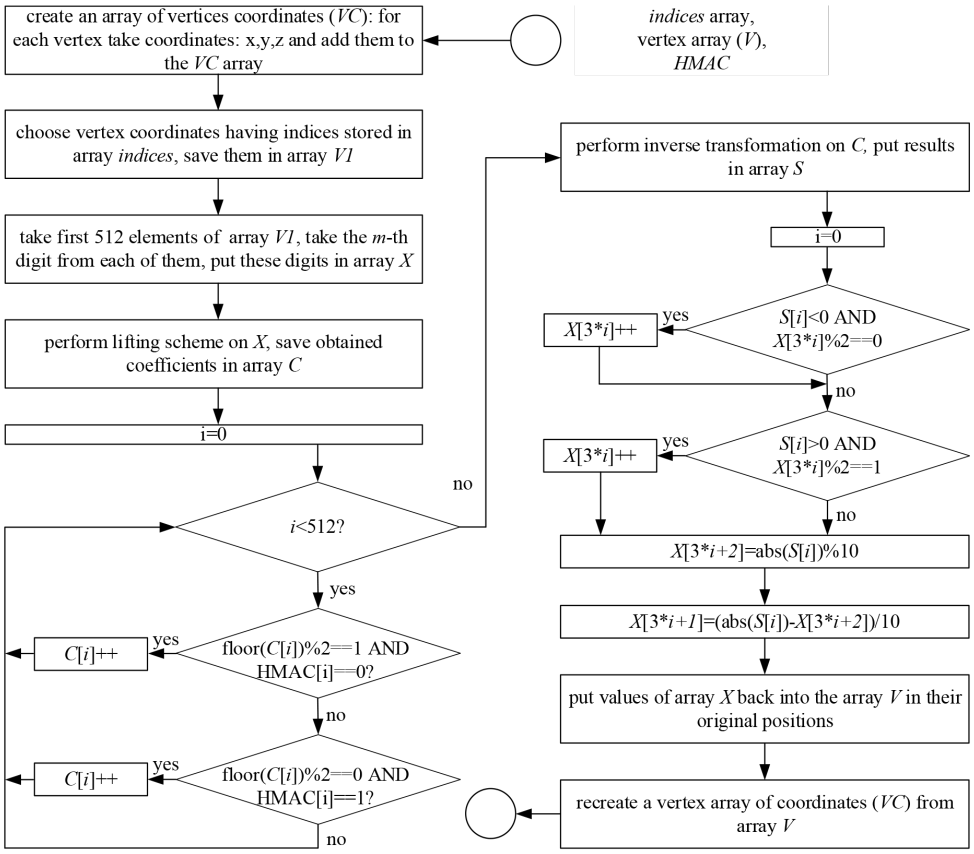


**Figure 2.** Scheme of a fingerprint attachment

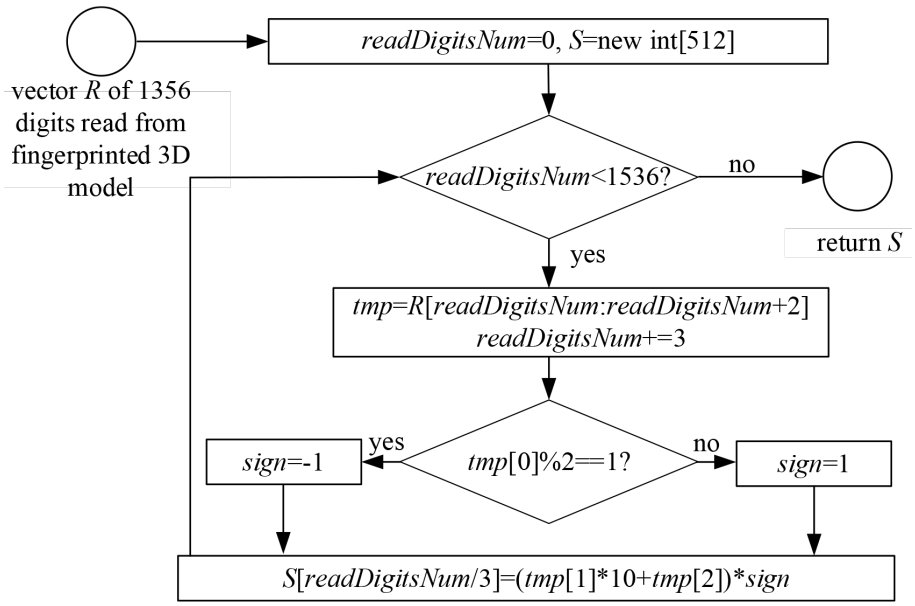


**Figure 3.** Scheme of the *S* array decoding process

## Interference metrics

3D model fingerprinting involves modifying the position of selected vertices. This results in a change in the shape of the 3D object. We consider these changes as distortions. To make an objective assessment of the interference caused by fingerprinting, it is necessary to use distortion measures. Due to the nature of the modifications introduced, this paper uses such measures as peak

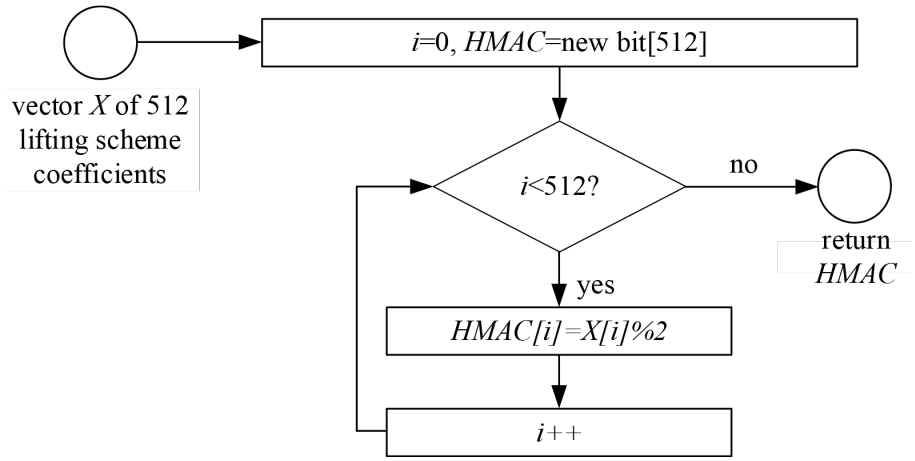


**Figure 4.** Scheme of the HMAC decoding process

signal-to-noise ratio (*PSNR*), mean square error (*MSE*) and structural similarity (*SSIM*) defined for 3D models. PSNR is a measure of distortions calculated according to the formula:

$$PSNR = 10 * log_{10} \frac{N*\vec{V}_{max}}{\sum_{i=1}^{N} \vec{V}_i^* - \vec{V}_i^2} \quad (6)$$

where: $N$ is the overall number of vertices, $\vec{V}_{max}$ is the most remote point from the centre of the model, $\vec{V}_i^*$ is the current vector in the fingerprinted model, and $\vec{V}_i$ is a corresponding vector in the original model. The bigger the PSNR, the less distorted is the model.

The MSE is a measure that describes the distance between points of the original model and the modified one. The smaller the MSE value, the less altered the model. It is calculated according to the formula:

$$MSE = \frac{\sum_{i=1}^{N} (\vec{V}_i^* - \vec{V}_i)^2}{N} \quad (7)$$

The root mean square error is calculated according to the formula which corresponds to the square root value of the MSE:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (\vec{V}_i^* - \vec{V}_i)^2}{N}} \quad (8)$$

Structural Similarity Index Measurement (SSIM) is a measure originally used to compare the structural similarity of images. It can also be utilised in 3D objects comparison, for example, operation on orthographic projections of the model. It is then calculated according to the formula:

$$SSIM(I_1, I_2) = \frac{(2*\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (9)$$

where: $\mu_x$ is the mean over a window in the original 3D image, $\mu_y$ is the mean over a window in the fingerprinted 3D image, $\sigma_x$ is standard deviation (square root of variance) over a window in the original 3D image, $\sigma_y$ is standard deviation (square root of variance) over a window in the fingerprinted 3D image, $\sigma_{xy}$ is co-variance over a window between analysed images, $x$ and $y$ refer to a local window in the analysed images, C1 and C2 are small constants as it is in Eq. (9) [49].

$L^2$ measurement is a metric describing the distance between surfaces in 3D space. It can be used to measure average distortion for all vertices with the following formula:

$$d(M, M') = \frac{1}{|M|} \sum_{i=1}^{|M|} |v_i - v_i'| \quad (10)$$

where: $M$ and $M'$ are original and watermarked models, and $v_i$ and $v_i'$ are positions of the $i^{th}$ vertices in the models [37].

## RESULTS AND DISCUSSION

### Interference introduced by model modifications

The proposed method attaches data by modifying the position of the model's vertices. It is therefore important to evaluate the effect of the modifications made on the distortion of the model. Since the coordinates of the vertices are represented by numbers written with a precision of seven decimal places and hiding consists of modifying the values of selected digits of these coordinates,

a study was conducted to determine the amount of distortion caused by modifying the digits at the $n^{th}$ decimal place in all coordinates and 1536 selected coordinates. The level of distortion introduced by modifying 1536 digits at different decimal places was also determined. This study aims to determine the optimal selection of digits for modification allowing to minimise the distortion introduced into the fingerprinted model. In this examination selected digits were converted to random values. The results obtained are shown in Table 2.

The results presented in Table 2 showed that as the significance of the modified digit increases, the level of interference introduced rises. From the proposed method's point of view, it is interesting to see what effect modifying the 1536 coordinates has on the level of distortion. The results obtained revealed that modifying the digits from the third decimal place onwards causes negligible distortion (marked bold in Table 2). They reached a PSNR under 50dB for all models tested. Modification of two digits in each coordinate that reduces the number of modified coordinates did not significantly affect distortion levels. Models with modified digits at 6 and 7 decimal positions in 768 vertices achieved results that were between those achieved for models with one digit modified in 1536 vertices at 7 and 6 decimal positions (marked grey in Table 2). This makes it clear that it is possible to hide the fingerprint by modifying the digits at the $n^{th}$ and subsequent decimal positions in the coordinates, and that this will not introduce greater distortions than would be introduced if the fingerprint were hidden by modifying the 1536 digits at the $n^{th}$ decimal position.

The proposed fingerprinting method also has implications for the distortion of components produced from fingerprinted models. The basic unit of measurement in the analysed glb format files is the metre. This means that a value of 1 in the vertex coordinate corresponds to one metre. Modifying the digit in the vertex coordinate at the seventh decimal place by one causes the vertex to be shifted on the printout by 0.0001mm. A maximum modification of a digit at the seventh decimal place shifts the vertex on the printout by 0.0009 mm. Model fingerprinting will result in up to 1536 model vertices being shifted. For 3D printing, such distortions are usually irrelevant, because the precision of printers is around 0.1mm. Resin printers can achieve precisions of about 0.025 mm. This means that in practice, a change of 0.0009 mm is likely too small to be physically realised in the printed model, as it's below the precision threshold of most 3D printers.

When modifying digits at the sixth decimal place, the loss of precision of the model can be up to 0.009mm and for digits at the fifth decimal place up to 0.09mm which can be a problem when dealing with high-precision parts.

**Table 2.** Interference level introduced by modification of chosen digits in vertices coordinates as PSNR

| Modified digits | PSNR [dB] | | | | | |
|---|---|---|---|---|---|---|
| | tree | pinecone | sculpture | gate | brick | dinosaur |
| 7th decimal place in all coordinates | 167.6 | 124.9 | 138.0 | 157.7 | 149.7 | 159.3 |
| 6th decimal place in all coordinates | 129.7 | 105.8 | 119.0 | 143.6 | 136.3 | 142.2 |
| 5th decimal place in all coordinates | 109.5 | 85.8 | 99.0 | 125.1 | 116.4 | 122.2 |
| 4th decimal place in all coordinates | 89.5 | 65.8 | 79.0 | 105.1 | 96.3 | 102.5 |
| 3rd decimal place in all coordinates | 69.5 | 45.8 | 59.0 | 85.1 | 76.3 | 83.2 |
| 2nd decimal place in all coordinates | 49.4 | 25.8 | 39.0 | 65.1 | 56.3 | 62.5 |
| 1st decimal place in all coordinates | 49.3 | 21.9 | 29.8 | 45.1 | 36.3 | 42.4 |
| 7th decimal place in 1536 coordinates | **158.9** | **132.0** | **147.3** | **150.5** | **155.0** | **148.3** |
| 6th decimal place in 1536 coordinates | **138.7** | **112.4** | **127.0** | **150.7** | **135.2** | **127.9** |
| 5th decimal place in 1536 coordinates | **119.5** | **92.5** | **107.6** | **131.1** | **115.6** | **108.3** |
| 4th decimal place in 1536 coordinates | **99.7** | **71.9** | **86.8** | **110.1** | **96.2** | **88.4** |
| 3rd decimal place in 1536 coordinates | **79.3** | **52.4** | **67.2** | **90.4** | **75.8** | **68.4** |
| 2nd decimal place in 1536 coordinates | 59.9 | 34.4 | 47.2 | 70.9 | 54.5 | 48.6 |
| 1st decimal place in 1536 coordinates | 39.8 | 16.1 | 30.1 | 50.5 | 37.0 | 26.2 |
| 7th and 6th dec. places in 768 coord. | **141.3** | **125.0** | **129.7** | **152.9** | **137.5** | **129.2** |
| 7th, 6th, 5th dec. places in 512 coord. | 122.7 | 96.5 | 111.4 | 134.7 | 118.8 | 110.1 |
| 2nd – 7th dec. places in 256 coord. | 66.3 | 40.6 | 55.6 | 78.4 | 60.4 | 54.5 |

## Model fingerprinting

The models shown in Table 1 were fingerprinted using the proposed method. For each of them, the values of the introduced distortions were calculated, and it was evaluated whether the originality of the model could be correctly verified. The results obtained are shown in Table 3.

## Change detection

To verify that the proposed method correctly detects modification of model data, a test was conducted by changing a randomly selected bit of the fingerprinted model. The method's ability to detect changes in the texture and position of the model's vertices was tested. The test of the ability to detect changes in vertex coordinates was divided into two independent tests. The first tested the ability to detect changes in digits that were not used to attach the fingerprint. In the second, the bits encoding the digits used to attach the fingerprint were modified. In addition, it was verified whether the method could detect modification of the order of vertices without changing the values of their coordinates. Each test was repeated 1,000 times and the results are shown in Table 4 as the percentage of correctly detected modifications in the test group.

The results showed that the proposed method can detect modifications introduced in any of the protected areas of the 3D model with 100% efficiency. The digits used to attach the fingerprint are also protected. Even though they are subject to modification and have not been included in the dataset from which the HMAC value is calculated, changes to their values are detected. This is because modification of any digits used to append the fingerprint causes a slight change in the transform coefficients, making it impossible to read the correct fingerprint.

The proposed method was finally validated by fingerprinting very large 3D scans of architectural objects made available as part of the work presented in [50, 51]. The method allowed seamless labelling of the models and the detection of the smallest changes made to them.

## Comparison to other solutions

The proposed fingerprinting method was compared with other solutions described in the literature. However, it should be emphasised that only a few solutions can confirm the originality of the 3D model and detect the change of a single bit of the model. Moreover, it is impossible to make an exact comparison, because each author uses different models of 3D objects in his work. In addition, the authors rarely use measures of distortion of 3D objects. Due to the lack of complete data on each method, the comparison was made based on available data.

**Table 3.** Performance of the proposed fingerprinting method

| Model | PSNR | RMSE | SSIM | $L^2$ | MSE | Originality verification |
|---|---|---|---|---|---|---|
| tree | 158.6 | $3.9 \times 10^{-4}$ | 99.6% | $9.0 \times 10^{-5}$ | $1.51 \times 10^{-7}$ | successful |
| pinecone | 131.9 | $7.39 \times 10^{-4}$ | 99.9% | $7.4 \times 10^{-4}$ | $5.46 \times 10^{-7}$ | successful |
| sculpture | 147.3 | $6.08 \times 10^{-4}$ | 99.9% | $1.7 \times 10^{-4}$ | $3.7 \times 10^{-7}$ | successful |
| gate | 149.9 | $2.94 \times 10^{-4}$ | 99.9% | $5.3 \times 10^{-5}$ | $8.65 \times 10^{-8}$ | successful |
| brick wall | 154.2 | $1.2 \times 10^{-3}$ | 99.7% | $7.2 \times 10^{-4}$ | $1.45 \times 10^{-6}$ | successful |
| dinosaur | 148.3 | $1.45 \times 10^{-3}$ | 99.7% | $1.3 \times 10^{-3}$ | $2.09 \times 10^{-6}$ | successful |

**Table 4.** Efficiency of the proposed method in modifications detection

| Model / type of modification | Texture modification | Coordinate modification | Modified digit modification | Vertices swap |
|---|---|---|---|---|
| tree | 100% | 100% | 100% | 100% |
| pinecone | 100% | 100% | 100% | 100% |
| sculpture | 100% | 100% | 100% | 100% |
| gate | 100% | 100% | 100% | 100% |
| brick wall | 100% | 100% | 100% | 100% |
| dinosaur | 100% | 100% | 100% | 100% |

The methods presented in [32, 33] allow a watermark originality verification by analysing how many parts of the watermark were recovered. This means that the methods allow for a certain level of model changes and do not allow for the detection of minor modifications. It only operates on the 3D model data without considering its textures or offering to check its originality, the same as the method presented in [52]. In [34] an original watermark is needed during verification to compare with the extracted one. Moreover, the procedure of vertices modification is complex due to the necessity of modifying vertices coordinates in the correct order. The methods proposed in [33, 53] require an original watermark to verify the originality of the 3D model. It allows coarse detection of modified portions of the model but does not provide confidence in detecting small changes. It also doesn't allow to detect single bits modifications. In addition, this method does not have a steganographic key, which gives the possibility of falsifying the watermark and makes it easier to detect its presence. The solution described in [36] modifies the position of a significant number of vertices (38%-45%). For typical models consisting of thousands of vertices, the modifications will involve a much larger number of vertices than the proposed method (1536), resulting in a significantly higher level of introduced distortion. Authors of [36] use the average distance (AD) as a distortion measure. It is calculated as the average distance between original and modified models' vertices distance. For the model examined in [36], authors declare AD ranging from $10^{-5}$ to 0,25 depending on used parameters' values. The fingerprinting with the proposed method results in AD=$10^{-5}$. The method allows the detection of model distortions and their locations. The authors do not specify how significant the distortions must be to be detected. However, the design of the algorithm, which examines the distance of a vertex from the barycentre of its neighbours and verifies whether it is within a defined range, indicates that small changes will not be detected. The same arguments arise when analysing the method proposed in [40]. The method modifies the positions of all vertices. Interference visibility reduction is possible by genetic algorithm usage.

The numerically stable fingerprinting algorithm proposed in [38] allows for detecting modifications of single bits of model coordinates. However, this is done at the cost of significant distortion – the number of shifted vertices ranges from 20% to 31%. Quite smaller but still significant distortion is introduced by the method presented in [37], where 12 to 16 percent of vertices are used to embed the watermark. The method presented in [39] is dedicated to the protection of computer-aided plant design. The proposed method is semi-fragile, which means that it is not able to detect changes in single bits. In the method proposed in [35], the size of the detected changes depends on the quantisation step. The problem with the presented method is the necessity of having information about the gravity centre of the original model. The method presented in [54] ensures fully blind fingerprint verification and can survive some common attacks and reduction of model floating-point precision. That results in the possibility of the introduction of small changes that will not be detected. The best of the methods mentioned seems to be the one presented in [20]. It is characterised by a very small fingerprint size (128 bits) and low distortion level (PSNR from 299 to 317dB), but it gives false negatives during fingerprint verification. Moreover, it is more time-consuming than the proposed method because it calculates the hash value for each pair of vertices encountered in the perimeter.

In addition to the above arguments, it should be noted that none of the presented algorithms take into account the evaluation of the texture integrity of the 3D model. Some of the authors mention such a possibility, but this problem was not developed in any of the cited works.

While this comparison provides insights into the relative strengths and weaknesses of various methods, it should be interpreted cautiously due to the aforementioned limitations. A more rigorous comparison would require standardized testing conditions and complete data across all methods.

## CONCLUSIONS

There are few methods for fragile fingerprinting of 3D models. It is crucial to ensure an efficient mechanism of 3D model authenticity verification to avoid falsification or unintended modifications in valuable 3D models that cannot be modified (3D models for printing, heritage scans, motion capture recordings). Most authors are still focused on ensuring robustness and allowing for some object modifications. There is a need for a tool for model authenticity and integrity verification that ensures a very low level of interference and allows the detection of single bit change.

Moreover, methods presented in the literature are focused on the geometrical shape of a model. It leaves a gap for texture verification. The method proposed in the paper fills the gap and offers the method for a 3D model authenticity check. It can detect a change of a single bit in model geometry or texture, making it possible to detect any changes to the model. A stegokey allows for spreading a fingerprint along the whole 3D model avoiding modification of many vertices in one region. A low level of introduced interference guarantees invisibility of the introduced changes and does not affect the model. The number of modified coordinates is constant independent of the model size. This makes the large models fingerprinting computationally efficient. The proposed method is reliable and does not give false positives or negatives as it happens in some other methods.

Future studies of the method will focus on minimising the introduced distortion by appropriately selecting the digits to be modified, so that the introduced value changes are minimised.

The issue of minimising the distortion introduced into the model is an open research problem. The authors plan to continue research in two directions. The first is the appropriate selection of digits for modification, so that the changes in values introduced are minimised. The second is to add functionality that allows modifications to be made to designated parts of the model, which will enable fingerprinting of models containing parts that cannot be distorted. A separate research issue is the applicability of the proposed method in augmented reality and virtual reality.

## Acknowledgements

The authors declare equal contribution to the paper. The authors are grateful to the Lublin University of Technology for the financial support granted to cover the publication fees of this research article.

## REFERENCES

1. Ligon S. C., Liska R., Stampfl J., Gurr M., Mülhaupt R. Polymers for 3D Printing and Customized Additive Manufacturing. Chemical Reviews, 2017, 117(15), 10212-10290. https://doi.org/10.1021/acs.chemrev.7b00074
2. Bagheri A., Jin J. Photopolymerization in 3D Printing. ACS Applied Polymer Materials, 2019, 1(4), 593-611. https://doi.org/10.1021/acsapm.8b00165
3. DebRoy, T., et al. (2018). Additive manufacturing of metallic components – Process, structure and properties. Progress in Materials Science, 92, 112-224. https://doi.org/10.1016/j.pmatsci.2017.10.001
4. Chen Z., Li Z., Li J., Liu C., Lao C., Fu Y., Liu C., Li Y., Wang P., He Y. 3D printing of ceramics: A review. Journal of the European Ceramic Society, 2019, 39(4), 661-687. https://doi.org/10.1016/j.jeurceramsoc.2018.11.013
5. Liu Z., Zhang M., Bhandari B., Wang Y. 3D printing: Printing precision and application in food sector. Trends in Food Science & Technology, 2017, 69(Part A), 83-94. https://doi.org/10.1016/j.tifs.2017.08.018.
6. Barszcz M., Dziedzic K., Skublewska-Paszkowska M., Powroznik P. 3D scanning digital models for virtual museums. Computer Animation and Virtual Worlds 2023, 34(3-4), e2154. https://doi.org/10.1002/cav.2154.
7. Skublewska-Paszkowska M., Powroźnik P., Barszcz M., Dziedzic K. Dual Attention Graph Convolutional Neural Network to Support Mocap Data Animation. Advances in Science and Technology Research Journal 2023, 17(5), 313-325. https://doi.org/10.12913/22998624/171592.
8. Nowomiejska K., Powroźnik P., Skublewska-Paszkowska M., Adamczyk K., Concilio M., Sereikaite L., Zemaitiene R., Toro M. D., Rejdak R. Residual Attention Network for distinction between visible optic disc drusen and healthy optic discs. Optics and Lasers in Engineering 2024, 176, https://doi.org/108056. 10.1016/j.optlaseng.2024.108056.
9. Józwik J., Dziedzic, K. Digital shape and geometric dimension analysis of polymer fuel tanks. *Advances in Science and Technology Research Journal* 2021*, 15*(4), 38-48. https://doi.org/10.12913/22998624/141649.
10. Montusiewicz J., Barszcz M., Dziedzic K. Photorealistic 3D digital reconstruction of a clay pitcher. *Advances in Science and Technology Research Journal* 2019*, 13*(4), 255-263. https://doi.org/10.12913/22998624/113276.
11. Peng F., Liao T., Long M., Li J., Zhang W., Zhou Y. Semi-Fragile Reversible Watermarking for 3D Models Using Spherical Crown Volume Division. IEEE Transactions on Circuits and Systems for Video Technology, 2023, 33(11), 6531-6543. https://doi.org/10.1109/TCSVT.2023.3272955
12. Modigari N., Valarmathi M., Anbarasi L. J., Gandomi A. H. Levenberg–Marquardt deep neural watermarking for 3D mesh using nearest centroid salient point learning. Scientific Reports, 2024, 14(1), 1-18. https://doi.org/10.1038/s41598-024-57360-z
13. Sola A., Sai Y., Trinchi A., Chen S. How Can We Provide Additively Manufactured Parts with a

Fingerprint? A Review of Tagging Strategies in Additive Manufacturing. Materials, 2022, 15(1), 85. https://doi.org/10.3390/ma15010085

14. Pham G. N., Lee S.-H., Kwon O.-H., Kwon K.-R. A 3D Printing Model Watermarking Algorithm Based on 3D Slicing and Feature Points. Electronics, 2018, 7(2), 23. https://doi.org/10.3390/electronics7020023
15. Wei C., Sun Z., Huang Y., Li L. Embedding anti-counterfeiting features in metallic components via multiple material additive manufacturing. Additive Manufacturing, 2018, 24, 1-12. https://doi.org/10.1016/j.addma.2018.09.003
16. Matvieieva N., Neupetsch C., Oettel M., Makdani V., Drossel W.-G. A novel approach for increasing the traceability of 3D printed medical products. Current Directions in Biomedical Engineering, 2020, 6(3), 315-318. https://doi.org/10.1515/cdbme-2020-3081
17. Xiao K., Chen X., Xu G., Xiao S., Yu N. A survey on 3D printing security: From concepts to applications. IEEE Internet of Things Journal, 2022, 9(11), 8424-8446.
18. Kozieł G. Fourier transform based methods in sound steganography. *Actual Problems of Economics* 2011*, 120*(6), 321-328.
19. Kozieł G. Simplified steganographic algorithm based on Fourier transform. *Advanced Science Letters* 2014, 20(2), 505-509. https://doi.org/10.1166/asl.2014.5322.
20. Botta M., Cavagnino D., Gribaudo M., Piazzolla P. Fragile watermarking of 3D models in a transformed domain. *Applied Sciences* 2020*, 10*(9), 3244. https://doi.org/10.3390/app10093244.
21. Ohbuchi R., Masuda H., Aono M. Watermarking three-dimensional polygonal models through geometric and topological modifications. *IEEE Journal on Selected Areas in Communications* 1998*, 16*(4), 551-560. https://doi.org/10.1109/49.668977.
22. Praun E., Hoppe H., Finkelstein A. Robust mesh watermarking. *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99,* 1999.
23. Ohbuchi R., Takahashi S., Miyazawa T., Mukaiyama A. Watermarking 3D polygonal meshes in the mesh spectral domain. In *Proceedings of the Graphics Interface 2001 Conference,* 2001.
24. Ohbuchi R., Mukaiyama A., Takahashi S. A frequency-domain approach to watermarking 3D shapes. *Computer Graphics Forum* 2002*, 21*(3), 373-382. https://doi.org/10.1111/1467-8659.00597.
25. Barni M., Bartolini F., Cappellini V., Corsini M., Garzelli A. Digital watermarking of 3D meshes. *Mathematics of Data/Image Coding, Compression, and Encryption VI, with Applications* 2004, 5208, 68–79. https://doi.org/10.1117/12.507437.
26. Yu Z., Ip H. H., Kwok L. A robust watermarking scheme for 3D triangular mesh models. *Pattern Recognition* 2003*, 36*(11), 2603–2614. https://doi.org/10.1016/S0031-3203(03)00158-1.
27. Li L., Zhang D., Pan Z., Shi J., Zhou K., Ye K. Watermarking 3D mesh by spherical parameterization. *Computer Graphics* 2004*, 28*(7), 981–989. https://doi.org/10.1016/j.cag.2004.07.004.
28. Cho J. W., Prost R., Jung H. Y. An oblivious watermarking for 3-D polygonal meshes using distribution of vertex norms. *IEEE Transactions on Signal Processing* 2007*, 55*(1), 142–155. https://doi.org/10.1109/TSP.2006.885729.
29. Mehta S., Prabhakaran B. 3D content fingerprinting. *2014 IEEE International Conference on Image Processing (ICIP), 2014,* 4797-4801. https://doi.org/10.1109/ICIP.2014.7025929.
30. Abdallah E. E., Abdallah A. E. Normal vectors and spanning tree for 3D object fingerprinting. *2013 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), 2013*. https://doi.org/10.1109/AEECT.2013.671646.
31. Yeo B.-L., Yeung M. M. Watermarking 3D objects for verification. *Proceedings of the 1999 IEEE International Conference on Image Processing (ICIP'99),* 1999, 1, 580-584. https://doi.org/10.1109/ICIP.1999.821608.
32. Wang Y. P., Hu S. M. A new watermarking method for 3D models based on integral invariants. *IEEE Transactions on Visualization and Computer Graphics* 2009*, 15*(2), 285–294. https://doi.org/10.1109/TVCG.2008.92.
33. Yeung M., Yeo B. L. Fragile watermarking of three-dimensional objects. *Proceedings of the International Conference on Image Processing,* 1998*,* 2, 442–446. https://doi.org/10.1109/ICIP.1998.723485.
34. Yeo B. L., & Yeung, M. M. Watermarking 3D objects for verification. *IEEE Computer Graphics and Applications* 1999*, 19*(1), 36–45. https://doi.org/10.1109/38.736467.
35. Huang C. C., Yang Y. W., Fan C. M., Wang J. T. A spherical coordinate based fragile watermarking scheme for 3D models. *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems,* 2013, 566–571. https://doi.org/10.1007/978-3-642-38577-3_66.
36. Chou C. M., Tseng D. C. A public fragile watermarking scheme for 3D model authentication. *Computer-Aided Design* 2006*, 38*(11), 1154–1165. https://doi.org/10.1016/j.cad.2006.07.001.
37. Chou C. M., Tseng D. C. Affine-transformation-invariant public fragile watermarking for 3D model authentication. *IEEE Computer Graphics and Applications* 2009*, 29*(2), 72–79. https://doi.org/10.1109/MCG.2009.20.
38. Wang, W. B., Zheng, G. Q., Yong, J. H., & Gu, H.

J. (2008). A numerically stable fragile watermarking scheme for authenticating 3D models. *Computer-Aided Design, 40*(6), 634–645. https://doi.org/10.1016/j.cad.2008.03.002

39. Su Z., Li W., Kong J., Dai Y., Tang W. Watermarking 3D CAD models for topology verification. *Computer-Aided Design* 2013*, 45*(8), 1042–1052. https://doi.org/10.1016/j.cad.2013.04.005.
40. Motwani M., Motwani R., Harris J. F. Fragile watermarking of 3D models using genetic algorithms. *Journal of Electronic Science and Technology* 2010*, 8*(3), 244–250.
41. Setiadi D. I. M., Rustad S., Shidik G. F. Digital image steganography survey and investigation (Goal, assessment, method, development, and dataset). *Signal Processing* 2023*, 206*. https://doi.org/10.1016/j.sigpro.2023.108476.
42. Wani M. A., Sultan B. Deep learning based image steganography: A review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2022, e1481. https://doi.org/10.1002/widm.1481.
43. Meng L., Xinghao J., Tanfeng S. A review of coverless steganography. Neurocomputing 2023. 566. 126945. https://doi.org/10.1016/j.neucom.2023.126945.
44. Boukhennoufa N., Laamari Y., Benzid R. Signal denoising using a low computational translationin-variant-like strategy involving multiple wavelet bases: application to synthetic and ECG signals. Metrology and Measurement Systems, 2024, 31(2), 259-278. https://doi.org/10.24425/mms.2024.148548
45. Zhang Z. The Improvement of the Discrete Wavelet Transform. Mathematics, 2023, 11(8), 1770. https://doi.org/10.3390/math11081770
46. Li, B., Liao, Y., Guo, R., Li, Z., Gu, Z.,Dai, X. Wavelet Transform and Damped Recursive Least Squares Method for Measurement Uncertainty Evaluation in EV Charging Pile Meters. Metrology and Measurement Systems, 2024, 31(3). https://doi.org/10.24425/mms.2024.150286
47. Hannoun K., Hamiche H., Lahdir M., Megherbi O., Laghrouche M., Bettayeb M., Robust digital image watermarking scheme with a fractional-order discrete-time chaotic scheme and DWT-SVD transform. Physica Scripta 2024, 99(5),055255. doi:10.1088/1402-4896/ad3d41.
48. Sweldens W. Wavelets and the Lifting Scheme: A 5 Minute Tour, https://cm-bell-labs.github.io/who/wim/papers/iciam95.pdf (Accessed: 2024.06.07).
49. Structural Similarity Index. https://www.ni.com/docs/en-US/bundle/ni-vision-concepts-help/page/structural_similarity_index.html (Accessed: 2024.06.07).
50. Kęsik J., Miłosz M., Montusiewicz J., Samarov K. Documenting the geometry of large architectural monuments using 3D scanning – the case of the dome of the Golden Mosque of the Tillya-Kori Madrasah in Samarkand. Digital Applications in Archaeology and Cultural Heritage 2021, 22, 1-11. https://doi.org/10.1016/j.daach.2021.e00199.
51. Miłosz M., Kęsik J., Montusiewicz J. Three-Dimensional Digitization of Documentation and Perpetual Preservation of Cultural Heritage Buildings at Risk of Liquidation and Loss—The Methodology and Case Study of St Adalbert's Church in Chicago. Electronics 2024, 13(3), 1-26. https://doi.org/10.3390/electronics13030561.
52. Chou C. M., Tseng D. C. Affine-Transformation Invariant Public Fragile Watermarking for 3D Model Authentication. *IEEE Computer Graphics and Applications* 2009, *29*(2), 72-79. https://doi.org/10.1109/mcg.2009.20.
53. Wu H.T., Cheung Y.M. A Fragile Watermarking Scheme for 3D Meshes. Proceedings of the 7th Workshop on Multimedia and Security, 2005, 117-124. https://doi.org/10.1145/1073170.1073192.
54. Lin H.Y.S., Liao H.Y.M., Lu C.S., Lin J.C. Fragile watermarking for authenticating 3-D polygonal meshes. IEEE Transactions on Multimedia 2005, 7(6), 997-1006. https://doi.org/10.1109/TMM.2005.858412.