Robust and quality-preserving 4D data watermarking for copyright protection

Ertugrul Gul[1,2*], Ahmet Nusret Toprak[3] and Gary K.L. Tam[2]

[1*]Software Engineering, Kayseri University, Kayseri, 38280, Turkey.
[2]Computer Science, Swansea University, Swansea, SA1 8EN, United Kingdom.
[3]Computer Engineering, Erciyes University, Kayseri, 38039, Turkey.

*Corresponding author(s). E-mail(s): ertugrulgul@kayseri.edu.tr;
Contributing authors: antoprak@erciyes.edu.tr; k.l.tam@swansea.ac.uk;

Abstract

The use of 4D (3D dynamic mesh sequences) data, such as rendering objects and faces, has grown significantly with advancements in imaging, 3D printing, and V/AR technologies. However, there is currently a lack of 4D watermarking methods to protect copyright and prevent unauthorized distribution. Existing 3D mesh watermarking techniques are inadequate for 4D data, as they fail to address dynamic attacks, such as sequence reordering. To fill this gap, we propose a novel 4D watermarking technique for 4D face data, comprising two main components: watermarking for 3D mesh sequences and watermarking for texture images. Each component uses two watermarks: one to protect individual mesh/-texture copyright and another to verify sequence order and detect reordering attacks. The Artificial Bee Colony algorithm optimizes embedding parameters to ensure watermark imperceptibility. Experimental results demonstrate the robustness and imperceptibility of our method, validating its effectiveness in protecting 4D data from unauthorized use and distribution.

Keywords: 4D watermarking, Robust watermarking, 4D face data, Artificial Bee Colony Algorithm, 3D mesh

1 Introduction

In recent years, technological advancements have led to exponential growth in the use, distribution, and sharing of multimedia data, including video [1], audio [2], and images [3–5]. However, this has also made it easier for unauthorized individuals to copy, modify, and distribute such data [6, 7]. Consequently, there is a critical need to protect multimedia content, copyright, and digital rights management globally [8–10]. Cryptography, steganography, and digital watermarking are three techniques used to protect digital content, each addressing different aspects of security and ownership [11]. Cryptography ensures confidentiality and integrity by converting data into unreadable formats; however, it can also reveal the existence of secret data through the ciphertext [12]. Steganography conceals communication by embedding secret information into media, such as images or videos, making the hidden message undetectable [13]. Digital watermarking embeds identifying information directly into multimedia content to assert ownership, trace distribution, or verify authenticity, focusing on robustness over secrecy [14]. While these methods may overlap in application, they have distinct purposes across various domains, such as data security, copyright protection, and covert communication.

Digital watermarking has emerged as a widely used and effective technique for safeguarding the copyright of digital data [15, 16]. The increasing use of 3D models (e.g., talking heads and facial capture) in various fields, such as video games, computer-aided design, 3D printing, and diagnostics, has heightened concerns about protecting their copyright and content [17]. To address this issue, researchers have been exploring 3D mesh watermarking methods. These methods embed hidden digital watermarks in 3D meshes without compromising their visual quality or functionality. The primary goal of 3D mesh watermarking is to ensure that the ownership and authenticity of 3D models can be verified by authorized parties, thereby preventing unauthorized distribution and usage.

3D mesh watermarking is a relatively new but essential field that has attracted the interest of many researchers. Various methods of 3D mesh watermarking have recently been developed to protect the copyright of a single 3D mesh and prevent unauthorized reproduction [18–20]. However, 3D meshes are often used in areas such as facial expression, speech, and movement detection, which require the processing of sequential 3D models. As a result, 4D data (which includes a series of 3D mesh and corresponding texture images) has become popular in recent years, with many 4D datasets used in studies [21, 22].

There are generally two categories of 3D mesh watermarking techniques [23]: *spatial* and *transform domain-based* techniques. Spatial domain techniques embed the watermark directly into the mesh structure, modifying elements like vertex coordinates, edge lengths, or triangular areas [24]. These modifications remain imperceptible to the human eye but can be detected by specialized methods. In contrast, transform domain techniques hide the watermark within the coefficients obtained through frequency, harmonic, or multi-scale transforms [24]. This involves using mathematical transformations like wavelets or Fourier transforms to extract the model's surface geometry information, where the watermark can be embedded without affecting the visual quality or functionality of the model. Overall, 3D watermarking is an essential tool for protecting the integrity and ownership of 3D models in various applications. Content creators can ensure their work remains secure and receive proper recognition and compensation for their efforts using these techniques.

Despite the popularity of utilizing 4D data for rendering objects and faces (esp., talking heads, facial capture), we observed that there is a lack of corresponding 4D watermarking methods that can safeguard copyright and prevent the unlawful distribution of 4D data. Most existing 3D mesh watermarking methods [20, 25–30] are not suitable for 4D data due to their inability to handle dynamic attacks like sequence reordering, which is crucial in 4D data. Currently, only two methods [31, 32] are available for dynamic 3D mesh watermarking. These methods assume detailed knowledge of vertex-to-vertex changes over time or require a fixed number of vertices (templates) in each 3D mesh frame. To obtain this information, they generally rely on post-processing techniques such as registration and template mesh fitting. However, these methods are not directly applicable to unprocessed data, which is typical of raw outputs from acquisition sensors where such information is not readily available. These methods also do not detect manipulations in sequence order. Additionally, watermarking only the 3D meshes within the 4D data is insufficient for security purposes, as it is also necessary to watermark the texture images of the 3D meshes.

One of the most challenging tasks for robust 3D mesh watermarking algorithms is achieving high imperceptibility and robustness simultaneously due to the trade-off between the two. Many 3D watermarking methods [18, 20, 25–30] rely on intuitive parameter determination, which may not be ideal as it significantly affects the method's performance. This approach also limits the applicability of watermarking since the optimal parameters may vary for each 3D mesh and watermark. In recent years, optimization algorithms have been successfully applied to 3D mesh watermarking to determine the optimal watermark embedding parameters [33–36]. However, although these methods consider imperceptibility and robustness in the optimization process, we observe that they do not guarantee a desired level of quality in the watermarked mesh. The optimization of embedding parameters should aim to achieve maximum robustness while ensuring that a predetermined quality threshold guarantees the imperceptibility of the watermarked 3D mesh with texture.

This paper introduces a novel 4D data watermarking method to overcome the limitations mentioned above and protect the copyright of 4D face data. To the best of our knowledge, this is the first study in the literature to directly address 4D data watermarking without relying on prior vertex-to-vertex or template information. The proposed method consists of two key components: 3D mesh sequence watermarking and texture image watermarking. Both components generate and embed two types of watermarks: sequence-level and mesh/image-level. The sequence-level watermark provides protection for each 3D mesh or texture image within the entire sequence. This type of watermark is the same across all frames in the sequence, serving as a consistent proof of copyright for each individual mesh or texture image. This means that even if unauthorized individuals obtain one or a few meshes or texture images from the sequence, the sequence-level watermark can still verify the copyright ownership of those individual pieces, linking them back to the original watermarked sequence. On the other hand, the mesh-level or image-level watermark is unique to each specific mesh or texture image. This type of watermark ensures that each frame is correctly ordered within the sequence, which helps in detecting any reordering attacks. Additionally, these watermarks provide an extra layer of protection by verifying the ownership of each mesh or image and safeguarding against various types of attacks at the mesh/image level.

We have developed a histogram equalization-based 3D mesh sequence watermarking algorithm and a Discrete Wavelet Transform (DWT) and Singular Value Decomposition (SVD)-based texture image watermarking method to create a robust and quality-preserving 4D data watermarking method for copyright protection. Both proposed methods utilize the Artificial Bee Colony (ABC) algorithm [37] to determine optimal embedding parameters. This ensures that the watermarked 3D meshes maintain acceptable visual quality, unlike existing methods.

The main novelty of our work is the development of the first watermarking approach specifically designed for 4D data, which includes both dynamic 3D mesh sequences and their corresponding texture images. Unlike existing 3D dynamic watermarking techniques, our proposed method does not require a fixed number of vertices in 3D mesh frames for watermarking. Furthermore, our approach integrates a perceptual quality threshold into the optimization-based watermark embedding process, ensuring an optimal balance between robustness and visual imperceptibility. To the best of our knowledge, this is the first work applying a quality-threshold constraint in 3D mesh watermarking, representing a significant advancement in the field of digital watermarking.

The key contributions of our work can be listed as follows:

- We introduce the first-ever watermarking method specifically designed for 4D data, including 3D mesh sequences and texture images, that does not require prior knowledge of vertex-to-vertex changes over time or a fixed number of vertices per frame.
- We propose a watermarking method for 3D mesh sequences that can guarantee the quality of the 3D meshes and can also detect changes in their order. To our knowledge, this is the first study that ensures imperceptibility, applying a quality threshold in 3D mesh watermarking.
- We also present a quality-ensured texture image watermarking method. This component protects the copyright of the texture images within 4D face data and can detect changes in their order.
- We have generated and publicly released a 4D face sequence dataset, providing a benchmark resource to facilitate future research and comparative studies in 4D watermarking.

The rest of the paper is structured as follows. Related works are given in Section 2. Section 3 describes our 4D face sequence data. Section 4 explains the proposed 4D watermarking method, while Section 5 presents the experimental results. Section 6 concludes the paper.

2 Related works

This section is organized into four subsections: single 3D mesh watermarking, dynamic 3D mesh watermarking, the quality threshold for watermarking, and an analytical summary of related works. The first two subsections cover methods for watermarking single and dynamic 3D meshes. The third focuses on optimization-based approaches, while the fourth provides an analytical comparison of related works, highlighting the relevance and contribution of our method to 4D watermarking.

2.1 Single 3D mesh watermarking methods

Researchers have proposed several 3D mesh watermarking techniques to protect the copyright of a single mesh. The most remarkable one is proposed by Cho et al. [18]. This method involves modifying the distribution of vertex norms to embed the watermark information into the single-cover 3D mesh. Each watermark bit is embedded into distinct sets called bins. This method has been used as the basis for many other techniques, making it a highly influential approach in the field. Medimegh et al. [25] proposed a 3D mesh watermarking technique using salient feature points, which are detected using an auto diffusion function-based salient point detector. The mesh is segmented according to the detected salient points, and a watermark is embedded into each region using Cho et al.'s method. Delmotte et al. [26] presented a blind watermarking method that is low-visibility for 3D-printed objects and robust against print scan attacks. This method is developed based on Cho et al.'s method and uses moment alignment and surface norm distribution. Local curvature estimation-based statistical watermarking method is proposed by Sharma and Panda [27]. In this method, vertices with moderate smoothness measure are selected using a feature vector, and the watermark is embedded utilizing the spread of vertex norm. The embedding process of this method is the basis of Cho et al.'s method. Medimegh et al. [28] presented a 3D mesh watermarking method based on non-negative matrix factorization. In this method, the mesh is segmented into parts using the salient point, and non-negative matrix factorization is applied to the vertex norms of parts. Then, the watermark is inserted into the bins of the extracted region using Cho et al.'s method. Modulation of radial distance distribution in the deeper surface-based 3D mesh watermarking method is proposed by Singh and Devi [20]. In this method, the watermark is embedded into selected deeper surface vertices by using the histogram mapping function suggested by Cho et al. Choi et al. [29] proposed a cropping-resilient synchronization-based 3D mesh watermarking method. In this method, mesh synchronization is applied using the basis point and scale information, and a watermark is embedded using the watermarking method, which is an improved version of Choi et al.'s method. Hu et al. [30] presented a robust 3D mesh watermarking method based on prongs for transmission security protection through sensor networks. The watermark is embedded into the neighborhood regions of the selecting prominent feature vertices using an extension of Cho et al.'s method. Hou et al. [38] presented a traitor tracing and access control methodology based on watermarking, wherein the watermark embedding technique is based on Cho et al.'s method. Alhammad et al. [39] proposed a novel 3D object watermarking approach that robustly embeds a grayscale image three times into the object's vertices using DWT. Hu et al. [40] present a robust 3D watermarking method that uses empirical mode decomposition, in which the watermark is embedded into the stable extreme points. The methods described above are designed for individual 3D meshes and can be applied separately to the 3D meshes within 4D datasets. However, they are unable to detect manipulations in the sequence order of 3D meshes, which is crucial for maintaining the temporal coherence and structural integrity of the 4D sequence. Additionally, since 4D data consists of both a 3D mesh sequence and associated texture images, protecting only the 3D meshes is insufficient to fully secure the 4D data. Furthermore, the 3D meshes in our 4D face data contain flat regions. As a result, methods based on Cho et al.'s embedding approach may face challenges, such as empty bins or bins with insufficient vertices for effective watermarking.

2.2 Dynamic 3D mesh watermarking methods

On the other hand, we noticed that out of all the techniques, only two are specifically designed to work with dynamic meshes. The first method is the robust watermarking approach proposed by Tsai et al. [32], which utilizes mesh segmentation and vertex trajectory. This method segments a mesh frame into parts using the shape diameter function. Independent sets are created based on geodesic distances between vertices and a cutting boundary. Watermark is embedded by adjusting the mean geodesic distance. Vertex trajectories modify the geodesic distribution of remaining frames for a watermarked animation. The second dynamic mesh model watermarking method is proposed by Kim et al. [31], which uses the distribution of temporal wavelet coefficients. This method applies the wavelet transform to each vertex coordinate having the same connectivity index along the temporal axis. The distribution of wavelet coefficients in frames with high or middle temporal frequency is then modified to watermark bits to be embedded. Both of these methods require that the number of vertices on the time axis is fixed and that the change of each vertex is known. However, these methods cannot be applied to 4D data that lacks a fixed number of vertices and has unknown trajectories. In such cases, existing methods are unsuitable, as they are not designed to handle this type of data. Therefore, new watermarking methods capable of managing unknown trajectories are needed for 4D data, such as our face data. Our study addresses this research gap.

2.3 Quality threshold for watermarking

Most 3D mesh watermarking algorithms, which use the strength factor, determine it intuitively. However, due to the trade-off between imperceptibility and robustness, manually determining the strength factor is challenging for these algorithms. Therefore, some researchers use the optimization algorithm to solve this problem. Tamane et al. [36] proposed a blind 3D mesh watermarking method based on DCT, Wavelets, and optimization. To determine the optimal strength factor for the watermarking, they employed a Constrained Nonlinear Optimization approach. However, the optimization process only took into account the imperceptibility of the watermarked mesh and utilized the peak signal-to-noise ratio as the fitness function. Unfortunately, this method did not provide any consideration for the robustness or balance between robustness

Table 1: Summary and comparison of related works.

Method	3D mesh	3D Dynamic "template" mesh	4D (3D Dynamic mesh with texture image)
Cho et al. [18]	✓	○	×
Hu et al. [30]	✓	○	×
Choi et al.[29]	✓	○	×
Medimegh et al. [25]	✓	○	×
Delmotte et al. [26]	✓	○	×
Sharma and Panda [27]	✓	○	×
Medimegh et al.[28]	✓	○	×
Singh and Devi [20]	✓	○	×
Hou et al. [38]	✓	○	×
Alhammad et al. [39]	✓	○	×
Hu et al. [40]	✓	○	×
Kim et al. [31]	×	✓	×
Tsai et al. [32]	×	✓	×
Proposed Approach	○	○	✓

✓: Originally designed for this data type. ○: Also applicable to this data type,
×: Not applicable for this data type.

and imperceptibility. Mouhamed et al. [35] presented a genetic algorithm-based 3D mesh watermarking method. This algorithm utilizes the genetic algorithm (GA) to select the optimal parameter lambda, which is then employed to modify the statistical distribution to get an optimal balance between robustness and imperceptibility. In this method, the fitness function is created based on the weighted sum of the imperceptibility and robustness results. A 3D mesh watermarking method based on Ant Colony Optimization (ACO) and Weber Law is proposed by Narendra et al. [34]. The optimum strength factor, which is identified for embedding the watermark, used in this method is determined with the ACO algorithm. In this method, the multiplication of the imperceptibility result and robustness results is defined as the fitness function. Mouhamed et al. [33] proposed a 3D mesh watermarking method based on the Coyote Optimization Algorithm (COA), which is used to optimize the controlling parameter. In this method, the fitness function for optimization is formulated with a weighted sum of imperceptibility and robustness. Existing optimization-based methods aim to balance imperceptibility and robustness, typically using a fitness function focused on imperceptibility or a weighted sum of both. However, none incorporate a quality threshold for watermarking, which is essential for applications like medical imaging, animation, and virtual reality, where maintaining the visual integrity and functionality of 3D meshes is critical. Without a quality threshold, these methods cannot guarantee that the watermarked meshes will meet the necessary standards for these applications. In contrast, our method is the first to address this gap by incorporating a quality threshold, ensuring that watermarked 3D meshes retain both high quality and robustness.

2.4 Summary of comparison

We summarize our observations and the applicability of these methods to 4D data in Table 1. Existing 3D mesh watermarking methods, such as those proposed by Cho et al. [18] and others, primarily focus on watermarking individual 3D meshes. While these methods can watermark meshes in 4D data separately, they fail to detect changes in the sequence order, which is crucial for maintaining the integrity of the 4D data. Furthermore, these methods struggle with small models or those with flat regions, as some watermarking bins may be empty or lack enough vertices. Additionally, most of these approaches rely on intuitively selected strength factors and are not suitable for 4D data with unknown trajectories or variable vertex counts.

In contrast, our proposed method overcomes these challenges by incorporating both sequence-level and mesh-level or image-level watermarks, allowing for the detection of sequence order manipulations. It also introduces a quality threshold, a concept previously not applied to 3D mesh watermarking. Unlike existing methods, which often depend on manually selected parameters, our optimization-driven approach ensures that the watermarked 3D meshes meet a predefined quality threshold. This guarantees high-quality meshes, making our method ideal for applications where imperceptibility and mesh quality are critical.

3 4D face sequence dataset

The 4D recordings were generated using a 3dMD facial capture system [41], capturing sequences at 48 frames per second. This system employs infrared-based sensing technology, with two infrared cameras utilizing speckle patterns for geometry reconstruction, while the other four cameras are dedicated to capturing facial texture. Similar to other 4D capture systems, it generates data in the form of OBJ files for the 3D mesh and texture files for the facial surface, making this a typical setup for 4D data generation. We recruited a native Turkish speaker and captured the Turkish sentence ”Türkiye, dört mevsimin belirgin olarak yaşandığı bir ülkedir,” which translates to ”Turkey is a country where four seasons are experienced distinctly.” The recording lasted 7 seconds, resulting in a total of 433 frames. Each 3D frame contains approximately 3MB of mesh data and a 4MB texture image. The total data size for the 4D facial data sample, including the audio, is approximately 3 GB, with 1.55 GB for the texture files and 1.20 GB for the OBJ files. A sample frame from our capture is shown in Figure 1.

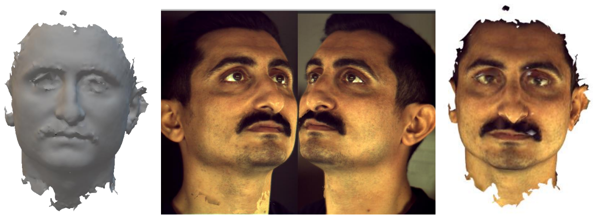


Fig. 1: Left: 3D mesh representation of a face; Center: corresponding texture image; Right: combination of 3D model-mesh and texture image

3D mesh in 4D facial data consists of a list of 3D coordinates, texture maps, and faces. The geometry of a 3D mesh is represented by vertices and faces. In the triangular mesh model, each vertex consists of the x, y, and z coordinates, $v_m = (v_x, v_y, v_z)$, while each face, $f = (v_i,\ v_j,\ v_k)$, consists of indices of the three vertices, forming a triangle shape as shown in Equation 1 [24]. Texture maps are coordinates that determine how the texture is applied to the mesh. The coordinates define 2D points with values between 0 and 1.

$$V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix},\ F = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} = \begin{bmatrix} v_{i1} & v_{j1} & v_{k1} \\ v_{i2} & v_{j2} & v_{k2} \\ \vdots & \vdots & \vdots \\ v_{in} & v_{jn} & v_{kn} \end{bmatrix} \tag{1}$$

The 4D Face Sequence Dataset has been specifically designed to support research in the area of 4D watermarking. Researchers can use this dataset to conduct experiments and assess the effectiveness of different 4D watermarking techniques. The dataset used in this study is publicly available on Zenodo and can be accessed via the DOI: 10.5281/zenodo.16621998.

4 Proposed 4D watermarking method

We propose an optimal, robust 4D data watermarking method, which ensures the quality of the 3D meshes and texture images. The proposed 4D watermarking method consists of two separate components: a 3D mesh sequence watermarking (Section 4.1) and a texture image watermarking (Section 4.2).

4.1 3D mesh sequence watermarking

In this section, we present an optimized 3D mesh sequence watermarking method tailored to address the unique challenges of protecting 4D data, particularly 3D face meshes. Our approach introduces two types of watermarks: sequence-level and mesh-level watermarks. The **sequence-level watermark** offers global protection for each mesh within the sequence. It ensures that copyright ownership can be verified even if individual meshes are extracted from the sequence. This watermark prevents unauthorized claims of ownership over single meshes, providing a proof of copyright for each piece. This ensures that even if unauthorized individuals obtain individual meshes, the sequence-level watermark can verify their copyright and link them to the original sequence. In contrast, the **mesh-level watermark** is unique to each mesh and safeguards the integrity of the mesh sequence by detecting and preventing temporal reorder attacks, where an attacker might manipulate the order of meshes in the sequence. By verifying the correct sequence of meshes, this watermark helps maintain the authenticity of the data.

Our method builds upon the framework introduced by Cho et al. [18], which embeds watermarks by modifying vertex norm distributions. However, Cho et al.'s approach has several limitations, notably its inability to handle reordering attacks and its lack of adaptability to diverse mesh geometries, especially for complex 3D face meshes with flat regions. To overcome these key technical challenges, our method integrates dynamic adjustments and optimizations, ensuring greater robustness and effectiveness. First, we modify the bin selection process from Cho et al.'s method [18] to better accommodate 3D face meshes with flat regions, which are often problematic in their approach. Our method dynamically adjusts the number of bins and ensures that each bin contains sufficient vertices for watermark embedding, eliminating the issue of empty or insufficient bins. Furthermore, we introduce the use of the Artificial Bee Colony (ABC) optimization algorithm to dynamically determine the optimal strength factor for each mesh sequence. This optimization process takes into account the unique characteristics of each mesh, ensuring that the watermark is both robust and imperceptible. By adapting the strength factor, our method achieves a balance between watermark imperceptibility and resilience to distortions or attacks.

The following sections outline the three key processes of the proposed method. Watermark Embedding (Section 4.1.1), Watermark Extraction (Section 4.1.2) and Optimization (Section 4.1.3).

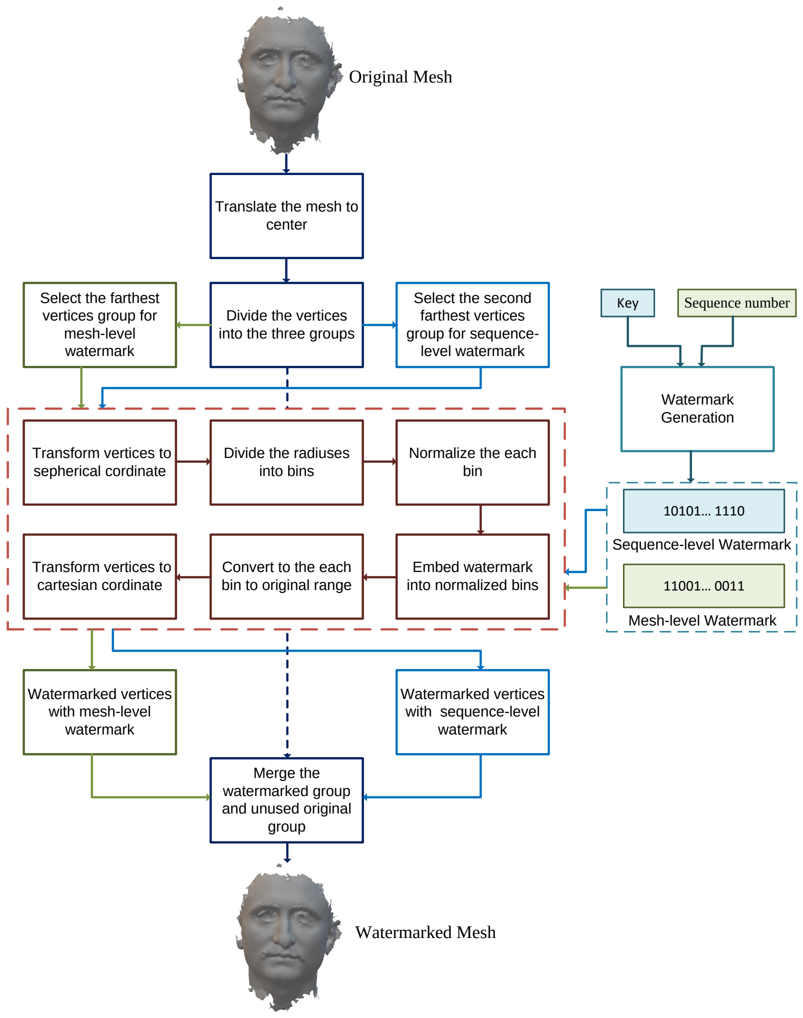
4.1.1 Watermark embedding process

The proposed method targets 4D facial data (Section 3). It generates mesh-level and sequence-level watermarks, which are then embedded into meshes. Firstly, the original mesh is translated to the center. Next, the vertices are divided into three groups based on their distance from the center. The group furthest from the center is used for mesh-level watermarking, while the middle group is used for sequence-level watermarking. The group of vertices closest to the center are not modified to preserve them due to their importance and meaning in the structure of our data (data close to the centre of the face).

To embed the mesh-level watermark, the vertices of the farthest group are transformed into spherical coordinates. The radiuses of the vertices are then selected and divided into bins. Normalization is applied to each bin, and a mesh-level watermark is embedded. Similarly, the sequence-level watermark is embedded into the vertices of the middle group using the same operations applied in mesh-level watermark embedding. Figure 2 shows the block diagram of the proposed embedding process for 3D meshes.

The core concept builds on Cho et al.'s method, which embeds watermark bits into a signal by adaptively adjusting the distribution of radii within each bin. A parameter, k_n, transforms the set of radii to make the mean of the transformed values approach 1/2. For a watermark bit of 1, the radii are adjusted so the mean slightly exceeds 1/2; for a bit of 0, the mean falls slightly below 1/2. This approach enables subtle yet robust watermark embedding while preserving the quality of the original signal. Our key innovation lies in the introduction of sequence-level and mesh-level watermarks in Step 2, which extends Cho et al.'s approach to address critical challenges. These enhancements ensure ownership at both levels and effectively counter-reordering attacks, which the original method cannot handle.

Detailed steps are provided below for clarity and reproducibility.



Original Mesh
Translate the mesh to center
Select the farthest vertices group for mesh-level watermark
Divide the vertices into the three groups
Select the second farthest vertices group for sequence-level watermark
Key
Sequence number
Watermark Generation
Transform vertices to sepherical cordinate
Divide the radiuses into bins
Normalize the each bin
10101... 1110
Sequence-level Watermark
Transform vertices to cartesian cordinate
Convert to the each bin to original range
Embed watermark into normalized bins
11001... 0011
Mesh-level Watermark
Watermarked vertices with mesh-level watermark
Watermarked vertices with sequence-level watermark
Merge the watermarked group and unused original group
Watermarked Mesh

Fig. 2: Embedding Process of 3D mesh watermarking

Step 1: Calculate the center, vc, of the mesh using Equation 2 and then translate every vertex in the mesh according to the center using Equation 3.

$$vc = \frac{\sum_j^N v_j}{N} \tag{2}$$

where v_j is jth vertex and N is the total number of vertices.

$$\begin{aligned} x_j &= x_j - x_{vc} \\ y_j &= y_j - y_{vc} \\ z_j &= z_j - z_{vc} \end{aligned} \tag{3}$$

where x_{vc}, y_{vc}, and z_{vc} denote the x, y, and z coordinates of the mesh center, vc, respectively.

Step 2: Divide the vertices of the meshes into three groups, $G_{gn}\quad (gn = 1,\ 2,\ 3)$. The division is applied according to the distance, d, of the vertices from the center, using Equation 4. The farthest one-third of the vertices is set for mesh-level watermarking, while the middle one-third is used for sequence-level watermarking. The vertices closest to the center are preserved.

$$d = \sqrt{((x_j - \ x_{vc})^2 \ + (y_j - \ y_{vc})^2 \ + \ (z_j - \ z_{vc})^2)} \tag{4}$$

Step 3: Select the farthest vertices group, G_1, of the mesh for mesh-level watermarking.

Step 4: Generate the mesh-level watermark, w^m, for mesh with secret key and mesh sequence number.

Step 5: Transform all vertices, $(x_j,\ \ y_j,\ \ z_j)$, in the group to spherical coordinate, $(r_j,\ \ \theta_j,\ \ \varphi_j)$.

Step 6: Divide the radiuses, r_j, into bins using Equation 5-7 and select the N non-empty bins (N is watermark size).

$$B_m = \{r_{m,\ k} \mid r_{m,\ min} < \ r < r_{m,\ max}\} \tag{5}$$

where B_m is mth bin, $r_{m,\ k}$ is the kth radius of the mth bin. $r_{m,\ min}$ and $r_{m,\ max}$ are the lower and upper bounds of the bin and they are calculated as follows:

$$r_{m,\ min} = r_{min} + \frac{r_{max} \ - \ r_{min}}{M}.m \tag{6}$$

$$r_{m,\ max} = r_{min} + \frac{r_{max} \ - \ r_{min}}{M}.(m+1) \tag{7}$$

where r_{min} and r_{max} are the lowest and highest radius, and M is the number of bins (both empty and non-empty bins).

Step 7: Normalize each selected bin to the range of [0, 1].

Step 8: Embed the n watermark bit into the normalized bins using Cho et al.'s method as follows:

Step 8.1: Select the first non-used bin, NB_n, and non-used watermark bit, NW_n.

Step 8.2: Transform the normalized radii in the bin, NB_n , by using Equation 8 and if $NW_n = 1$ go to step 8.3 if $NW_n = 0$ go to step 8.5.

$$r'_{n,\ k} = (r'_{n,\ k})^{k_n} \tag{8}$$

where k_n is a real number, which is initialized as 1.

Step 8.3: Calculate mean, μ_n, of transformed radii.

Step 8.4: If $\mu_n < (1/2) + \alpha$, decrease k_n ($k_n = k_n - \Delta k$) and go to step 8.2. otherwise, go to Step 8.7. Here, Δk represents the step size for incrementing or decrementing the parameter k_n, and α denotes the strength factor.

Step 8.5: Calculate the mean, μ_n, of transformed radii.

Step 8.6: If $\mu_n > (1/2){-}\alpha$, increase k_n ($k_n = k_n + \Delta k$) and go to step 8.2. Otherwise, go to the step 8.7.

Step 8.7: Apply Steps 8.1-8.7 until all watermark bits are embedded.

Step 9: Convert the radii of each bin to the original range.

Step 10: Transform the watermarked vertices, $(r'_j,\ \theta_j,\ \varphi_j)$, to cartesian coordinate, $(x'_j,\ y'_j,\ z'_j)$.

Step 11: Select the second group of vertices, G_2 for sequence-level watermarking and generate the sequence-level watermark, w^s, with a secret key.

Step 12: Apply steps 5-10 to embed the sequence-level watermark into the second group of vertices.

Step 13: Merge the watermarked groups of vertices with the non-modified group.

Step 14: Obtain the watermarked 3D mesh.

4.1.2 Watermark extraction process

In the proposed method for watermark extraction, each mesh is first translated to the center. Then, the vertices of each mesh are divided into three groups based on their distance from the center. The group of vertices farthest from the center is transformed into a spherical coordinate system, and bin division is performed on this group. Each bin is normalized within a range of [0, 1], and the mesh-level watermark is extracted from the radiuses of the normalized bins. Similarly, the sequence-level watermark is extracted from the vertices of the middle group using the same operations applied in mesh-level watermark extraction. Figure 3 shows the block diagram of the watermark extraction process for 3D mesh in the proposed method.

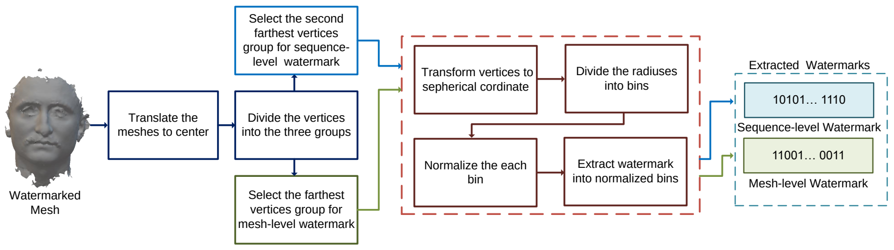

Watermarked Mesh
Translate the meshes to center
Select the second farthest vertices group for sequence-level watermark
Divide the vertices into the three groups
Select the farthest vertices group for mesh-level watermark
Transform vertices to sepherical cordinate
Divide the radiuses into bins
Normalize the each bin
Extract watermark into normalized bins
Extracted Watermarks
10101… 1110
Sequence-level Watermark
11001… 0011
Mesh-level Watermark

Fig. 3: Extracting process of 3D mesh watermarking

Here are the detailed steps of the proposed watermark extraction process for 3D meshes of 4D data:

Step 1: Calculate the center of the mesh, vc, using Equation 2 and then translate the mesh according to the center using Equation 3.

Step 2: Divide the vertices of the mesh into three group, G_{gn} $(gn = 1,\ 2,\ 3)$, based on their distance, d, from the center using Equation 4.

Step 3: Select the group of farthest vertices, G_1, in the mesh to extract the mesh-level watermark.

Step 4: Transform the vertices, $(x_j,\ \ y_j,\ \ z_j)$, to spherical coordinate, $(r_j,\ \ \theta_j,\ \ \varphi_j)$.

Step 5: Divide the radiuses, r_j, into bins using Equation 5- 7 and select the n non-empty bins, (n is watermark size).
Step 6: Normalize each selected bin to range [0, 1].
Step 7: Extract the mesh-level watermark, w^{em}, from the normalized bins using Cho et al.'s watermark extraction method as follows:
    Step 7.1: Select the first non-used bin, NB_n.
    Step 7.2: Calculate mean, μ_n, of the bin.
    Step 7.3: If $\mu_n > 1/2 \quad W_n = 1$ else $W_n = 2$
    Step 7.4: Apply the 7.1-7.3 to extract all watermark bits.
Step 8: Select the second group of vertices, G_2, from the mesh for sequence-level watermark extraction.
Step 9: Apply the Step 4-7 to extract the sequence-level watermark.
Step 10: Obtain the mesh-level watermarks, w^{em}, and sequence-level watermark, w^{es}.

4.1.3 Optimization process

The optimization process in this study builds on Gul and Toprak's method [42], using the Artificial Bee Colony (ABC) algorithm to determine the optimal strength factor for 3D mesh watermarking in a sequence. Unlike single-image watermarking, our method embeds two types of watermarks, sequence-level and mesh-level, across a sequence of 3D meshes, ensuring robust protection tailored to 4D data challenges. ABC is particularly suited for our optimization task because it handles highly nonlinear, non-differentiable functions without needing gradient information, unlike gradient-based methods. ABC's global search mechanism efficiently navigates the solution space, avoiding local optima. Additionally, its adaptability allows it to tailor the strength factor to the unique structure of each mesh, achieving a balance between robustness and imperceptibility.

In the optimization process, our goal is to determine the optimal strength factor that maximizes the robustness of the watermark while ensuring the imperceptibility of the cover data. Existing watermarking methods often rely on manually set parameters, which can be inefficient and suboptimal. To address this limitation, optimization algorithms are widely employed to enhance watermarking techniques, especially in images and videos. To the best of our knowledge, this is the first work to incorporate a quality threshold within an optimization algorithm for 3D and 4D mesh watermarking. To evaluate the imperceptibility and robustness of the watermarked meshes, we use the Peak Signal-to-Noise Ratio (PSNR) and Normalized Correlation (NC) metrics. The PSNR value assesses the quality of the watermarked 3D mesh and is calculated as follows [43]:

$$PSNR = 20log_{10}\left(\frac{D_{max}}{RMSE}\right) \tag{9}$$

where D_{max} is the diagonal distance of minimal cuboid bounding box [44], and RMSE stands for root-mean-squared error, which is computed as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(V_i^o - V_i^w)^2} \tag{10}$$

where N is the number of vertices, and V_i^o and V_i^w are vertices of original and watermarked meshes, respectively. The NC value is a measurement that calculates the similarity between the extracted watermark and the original embedded watermark, indicating the robustness of the methods and calculated as follows [42]:

$$NC = \frac{\sum_{i=1}^{C}\sum_{j=1}^{R} W_{i,j}W'_{i,j}}{\sqrt{\sum_{i=1}^{C}\sum_{j=1}^{R}(W_{i,j})^2}\sqrt{\sum_{i=1}^{C}\sum_{j=1}^{R}(W'_{i,j})^2}} \tag{11}$$

where $C \times R$ is the size of watermark, W and W' are original and extracted watermark.

During the optimization process, both mesh-level and sequence-level watermarks are embedded into a sequence of 3D meshes. The PSNR value of each watermarked mesh is calculated for each solution in each generation to evaluate the visual quality. If the PSNR value of a solution falls below the predefined threshold of 68, it is penalized. Previous studies have shown that a PSNR value of 68 or higher indicates good visual quality for the watermarked mesh [43, 45]. Subsequently, the robustness of the watermarked meshes is measured by subjecting them to a series of attacks. The NC value is calculated by extracting watermarks from the attacked meshes. These operations are repeated on each solution until a predefined condition is met, and the optimization process ends with the best solution obtained. Figure 4 displays the diagram of the optimization process.

The basic steps of the fitness value calculation of each solution for 3D meshes watermarking are given as follows:

Input: A candidate solution, $x_i = sf$, generated by the ABC algorithm and predefined threshold, Q_{th}, where sf is strength factor and x_i is ith solution. **Output:** Average NC value, NC_{avg}, of candidate solution x_i .

Step 1: Embed the mesh-level, W_j^m, and sequence-level, W^s, watermarks into all meshes, M_j^o, using the strength factor obtained from solution, x_i.

Step 2: Calculate the PSNR values, $PSNR_j$, of all watermarked meshes, M_j^w.

Step 3: If any of the PSNR value, $PSNR_j$, are lower than the predefined threshold, Q_{th}, then penalize the solution and return ϵ (a small positive number) to ABC algorithm as fitness value

Step 4: Apply n attacks to unprocessed watermarked mesh, M_j^w, and obtain the n attacked meshes, $M_{j,n}^A$.

Step 5: Extract the mesh-level, $W'^m_{j,n}$, and sequence-level, $W'^s_{j,n}$, watermarks from the attacked meshes, $M_{j,n}^A$.

Step 6: Calculate the average NC, $NC_{avg,i}$, of all extracted watermarks, $W'^m_{j,n}$ and $W'^s_{j,n}$.

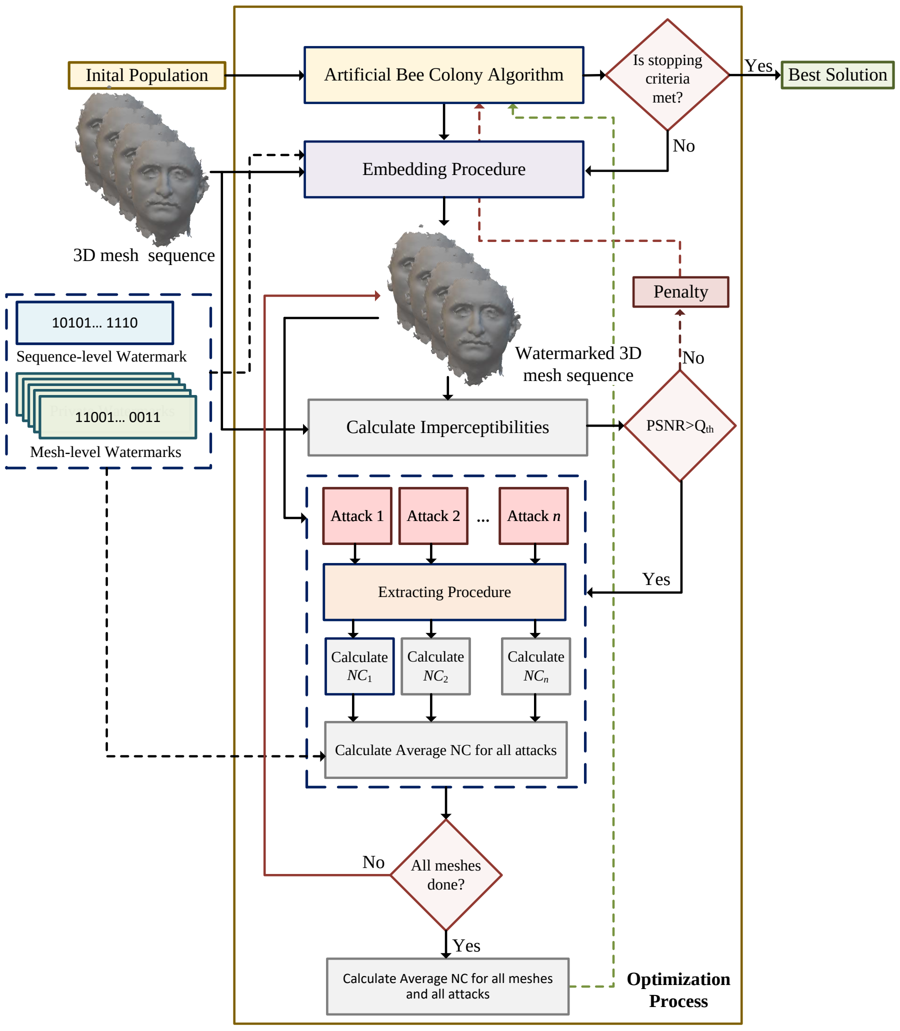


Fig. 4: Optimization process of 3D mesh watermarking

Step 7: Until all watermarked meshes, M_j^w, are processed, select the next unprocessed watermarked mesh, M_j^w, and go to Step 4.

Step 8: Calculate the average NC, NC_{avg}, of the average NC values, $NC_{avg,i}$, obtained from all meshes.

Step 9: Return average NC value, NC_{avg}, to ABC algorithm as fitness value.

During the optimization process, a number of attacks (n) are applied to assess the robustness of the proposed method (Step 4 in the calculation of fitness value) while determining the strength factor. The proposed algorithm allows the use of different numbers of attack types. In the experiments, we employed five distinct attacks: quantization, smoothing, subdivision, noise addition, and simplification, which are detailed in Section 5.1.

4.2 Texture images watermarking method

The proposed algorithm for embedding texture image watermarks is based on Gul and Toprak's method [42], which improves upon Tian et al.'s method [46]. In their study, Gul and Toprak improved the embedding equation and used the ABC algorithm to determine the strength factor and embedding positions. These methods were initially proposed for grayscale images. However, Gul and Toprak extended their work to embed the same watermark into the R, G, and B color channels to protect color images. Gul and Toprak determine the optimum strength factor and embedding positions for each image individually. However, in our case, we have a sequence of texture images in 4D data. Determining the optimal parameters for each image individually requires significant computational resources. Additionally, storing the optimal parameters for each image would also be necessary. It is also important to ensure consistency across the sequence.

We propose a texture image sequence watermarking method to protect the copyright of texture images in 4D face data. Similar to mesh watermarking, the proposed method uses two different types of watermarks: sequence-level and image-level. The sequence-level watermark is the same for all frames within the 4D face data, and it is embedded into texture images, providing robustness against manipulations. Even if an individual attempts to claim ownership by presenting a texture image derived from the model, the embedded watermark can be extracted and verified to establish true copyright ownership, linking back to the origin of the sequence. The image-level watermark serves two purposes: it ensures the correct order of texture images and offers additional copyright protection for the textured images within the 4D data.

In the texture image watermarking method, we employ the ABC algorithm to determine the optimal strength factor and embedding positions for all texture images in the 4D face data. The proposed method generates both image-level and sequence-level watermarks using a hash function, eliminating the need to store the embedded watermarks for later copyright verification. Additionally, we modify the watermark embedding algorithm proposed by Gul and Toprak by using two-level DWT instead of the Contourlet transform, as the watermark size in our case is smaller. We embed the image-level watermark in the R channel and the sequence-level watermark in the G channel, rather than embedding the same watermark across all channels. Preliminary experiments revealed that the R and G channels, which are more relevant to human faces, have a greater impact on the effectiveness of our image watermarking method.

Not all regions of the texture images are used to visualize the 4D face data, as shown in Figure 1. Specifically, certain areas, such as the left side of the left image and the right side of the right image, are used in the texture mapping. Our method offers flexibility in selecting regions for watermark embedding. We focus on the areas involved in the texture mapping, dividing the texture images into left and right parts, extracting their central regions, and then merging them for the watermarking process. Additionally, while the texture images in our 4D data may vary in size, our method is capable of handling images of different dimensions.

Similar to 4D watermarking, our texture watermarking comprises of three components: Watermark embedding (Section 4.2.1), extraction (Section 4.2.2), and optimization (Section 4.2.3).

4.2.1 Watermark embedding process

The proposed method begins by generating both image-level and sequence-level watermarks. The R channel of the texture image is used for embedding the image-level watermark, while the G channel is used for embedding the sequence-level watermark.

The original texture image is divided along the x-axis into left and right sections. From each section, a 512×256 region is extracted from the center, and these regions are merged to form a central image of size 512×512 pixels. This image is then converted to the RGB color space.

The R channel is divided into 32×32 blocks, and a two-level DWT is applied to each block. The DCT (Discrete Cosine Transform) and zigzag scan are performed on the LL2 sub-band of the blocks. From the zigzag-scanned DCT coefficients, eight coefficients are selected for each block. These coefficients are used to create two carrier matrices, on which SVD is performed. The largest singular values of these matrices are extracted, and the image-level watermark is embedded into them using the strength factor. A similar process is applied to the G channel to embed the sequence-level watermark, with the only difference being the channel and watermark type. Figure 5 shows the block diagram of the embedding process.

Detailed steps of the proposed watermark embedding process for texture images of 4D data are given as follows:

Step 1: Divide the original texture image, TIm^o, into left, LIm^o, and right, RIm^o, images. Then, determine the region for embedding watermarks. In our context, we use the centre parts, C^l and C^r, of the left and right images, LIm^o and RIm^o.

Step 2: Extract the 512×256 sized image's parts, Lp^o and Rp^o, from the left and right images using with centre points, C^l and C^r, as the midpoints.

Step 3: Merge the extracted 512×256 parts, Lp^o and Rp^o, to obtain central image, CIm^o, size of 512×512.

Step 4: Take the R, CIm^o_R, and G, CIm^o_G, channels of the central image.

Step 5: Generate the image-level, W^i, and sequence-level, W^s, watermark.

Step 6: Select the R channel, CIm^o_R, as a channel, Ch^o, select the image-level, W^i, watermark as a watermark, W.

Step 7: Divide the channel, Ch^o, into 32×32 blocks, Ch^o_i $(i = 1, \ldots, \; 512 \times 512/32 \times 32)$.

Step 8: Apply the 2 level DWT transform to 32×32 block, Ch^o_i, and take the LL2 sub-band, $LL2^o_i$.

Step 9: Apply the DCT transform to LL2 sub-bands, $LL2^o_i$.

Step 10: Apply the Zigzag scan to DCT coefficients, D^o_i.

Step 11: Select the eight coefficients of the Zigzag scanned DCT coefficients, Z^o_i, according to determined embedding positions.

Step 12: Generate the two carrier matrices, CM_1 and CM_2, and apply the SVD.

Step 13: Take the largest singular values, L_1 and L_2, of the two carrier matrices and calculate the average value of the largest singular values, L_{avg}.

Step 14: Embed the watermark bit, W_i, into the largest singular values, L_1 and L_2, using the strength factor, sf, as described in Equations 12 and 13, following the approach outlined in Section 4.1.

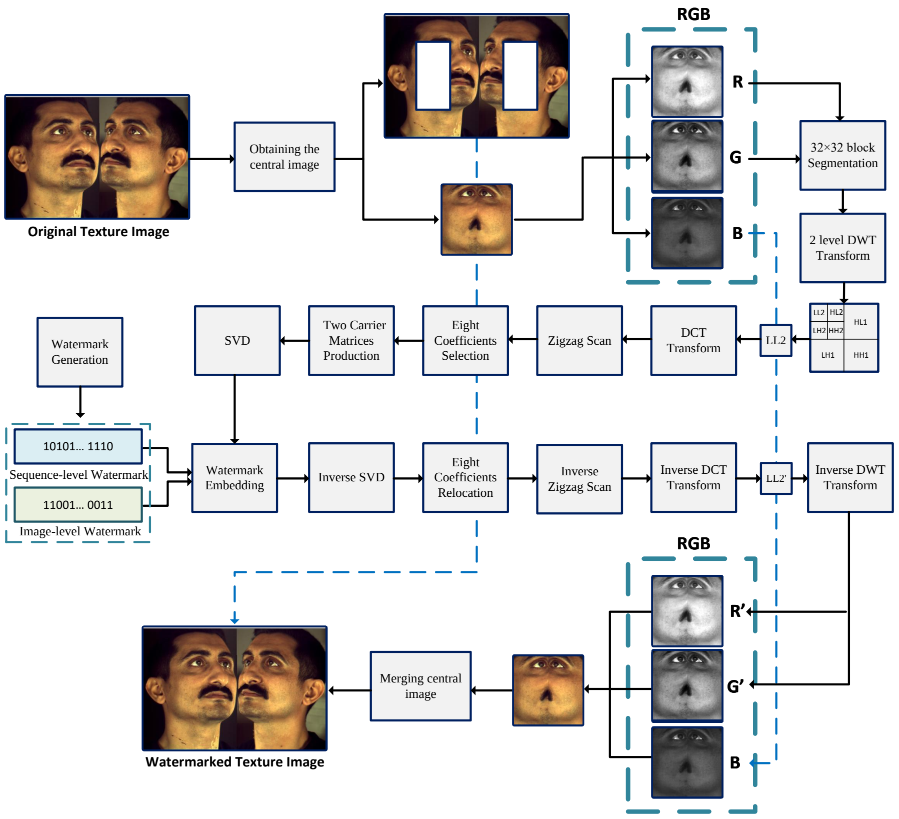


Fig. 5: Texture image watermark embedding process.

$$L'_1 = \begin{cases} L_{avg} + sf, & W == 1 \ \&\& \ L_1 - L_2 < 2 * sf \\ L_{avg} - sf, & W == 0 \ \&\& \ L_2 - L_1 < 2 * sf \\ L_1 & else \end{cases} \tag{12}$$

$$L'_2 = \begin{cases} L_{avg} - sf, & W == 1 \ \&\& \ L_1 - L_2 < 2 * sf \\ L_{avg} + sf, & W == 0 \ \&\& \ L_2 - L_1 < 2 * sf \\ L_2 & else \end{cases} \tag{13}$$

Step 15: Replace the watermark embedded largest singular values, $L\prime_1$ and $L\prime_2$, to original singular values, L_1 and L_2.

Step 16: Apply the inverse SVD to obtain two watermark embedded carrier matrices, CM'_1 and CM'_2, and replace to the originals, CM_1 and CM_2.

Step 17: Replace the watermark embedded 8 coefficients to originals and apply the inverse zigzag scan to watermarked zigzag scanned DCT coefficient, Z_i^w to get watermarked DCT coefficients, D_i^w.

Step 18: Apply the inverse DCT to watermarked DCT coefficient, D_i^w, to obtain the watermarked LL2 sub-band, $LL2_i^w$.

Step 19: Replace the watermarked LL2 sub-band, $LL2_i^w$, to original one and apply the inverse two level DWT to get watermarked block, Ch_i^w.
Step 20: Replace the watermark embedded 32×32 block, Ch_i^w, to original one, Ch_i^o.
Step 21: Apply the Step 8-20 for all 32×32 blocks, Ch_i^o.
Step 22: Select the sequence-level, W^s, as a watermark, W, and select the G channel, CIm_G^o, as a channel, Ch^o.
Step 23: Apply the steps 7-20 to embed the sequence-level watermark, W^s.
Step 24: Obtain the watermarked R, CIm_R^w, and G, CIm_G^w, channels and replace the original ones, CIm_R^o and CIm_G^o.
Step 25: Obtain the watermarked 512×512 central image, CIm^w.
Step 26: Divide the watermarked 512×512 central image, CIm^w, to 512×256 watermarked image parts, Lp^w and Rp^w.
Step 27: Replace the watermarked image parts, Lp^w and Rp^w, to left and right images according to centre points as the midpoints.
Step 28: Merge the watermarked Left and Right, LIm^w and RIm^w, images to obtain watermarked texture image, TIm^w.

4.2.2 Watermark extraction process

The proposed algorithm extracts a watermark from a watermarked texture image in the following series of steps. First, the image is divided along the x-axis into left and right images, and the center points of both sides are located. Next, 512×256 size parts of the image are extracted from both sides and merged to create a central image of size 512×512, which is then converted to RGB color space.

The R channel of the image is divided into blocks of size 32×32, and a 2-level DWT is performed on each block. DCT and zigzag scanning are applied to the LL2 sub-band of each block, and eight coefficients are selected from the zigzag-scanned DCT coefficients to create two carrier matrices per block. SVD is then performed on these matrices to determine the largest singular values for each block. The image-level watermark is extracted from the largest singular values of the two carrier matrices for each block. The same process is then repeated on the G channels to extract the sequence-level watermark. Figure 6 describes the block diagram of the entire extraction process in detail.

Detailed steps of the proposed watermark extraction process for texture images of 4D data are given as follows:
Step 1: Divide the watermarked texture image, TIm^W, into left, LIm^W, and right, RIm^W, images. Then, determine the centre, C^l and C^r, of the left and right images, LIm^W and RIm^W.
Step 2: Extract the 512×256 sized image's parts, Lp^W and Rp^W, from the left and right images using with centre points, C^l and C^r, as the midpoints.
Step 3: Merge the extracted 512×256 parts, Lp^W and Rp^W, to obtain central image, CIm^W, size of 512×512.
Step 4: Take the R, CIm_R^W, and G, CIm_G^W, channels of the central image.
Step 5: Select the R channel, CIm_R^W, as a channel, Ch^W.
Step 6: Divide the channel, Ch^W, into 32×32 blocks, Ch_i^W $(i = 1, \ldots,\ 512 \times 512/32 \times 32)$.

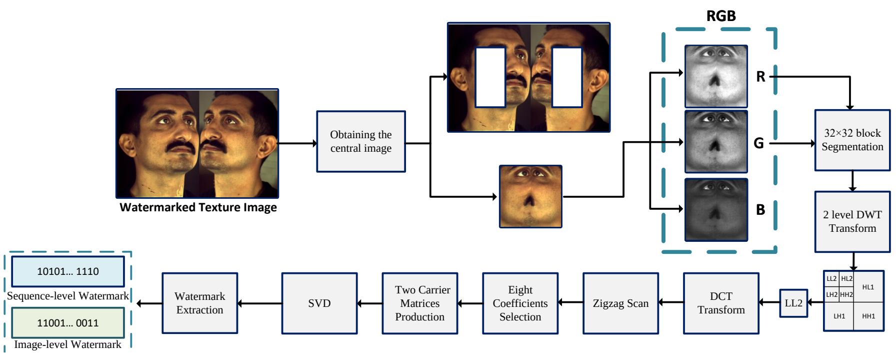


Fig. 6: Texture image watermark extraction process

Step 7: Apply the 2-level DWT transform to 32 × 32 block, Ch_i^W, and take the LL2 sub-band, $LL2_i^W$.
Step 8: Apply the DCT transform to LL2 sub-bands, $LL2_i^W$.
Step 9: Apply the Zigzag scan to DCT coefficients, D_i^W.
Step 10: Select the eight coefficients of the Zigzag scanned DCT coefficients, Z_i^W, according to determined embedding positions.
Step 11: Generate the two carrier matrices, CM_1^W and CM_2^W , and apply the SVD.
Step 12: Take the largest singular values, L_1^W and L_2^W, of the two carrier matrices.
Step 13: Extract the watermark bit, W'_i, from the largest singular values, L_1^W and L_2^W, according to Equation 14:

$$W'_i = \begin{cases} 1, & L_1^W \geq L_2^W \\ 0, & else \end{cases} \tag{14}$$

Step 14: Apply the step 7-13 for all blocks and get extracted watermark, W.
Step 15: Assign the extracted watermark as the image-level watermark, W_e^i.
Step 16: Select the G channel, CIm_G^W, as a channel, Ch^W.
Step 17: Apply the step 6-14 to get extracted watermark form G channel.
Step 18: Assign the extracted watermark as sequence-level watermark, W_e^s.

4.2.3 Optimization process

Our proposed method introduces a novel approach by utilizing an optimization procedure based on Gul and Toprak's method for texture image watermarking. Gul and Toprak's method searches for an optimum strength factor and embedding positions for each image to ensure robustness and imperceptibility. While Gul and Toprak's method embeds a single watermark into each image, our method generates different watermarks for each image in a sequence called image-level watermarks. To overcome the computational and memory requirements of optimizing embedding parameters for each individual texture image, we determine a single optimal parameter set for all images in the sequence. We further modified the embedding process of Gul and Toprak's method to accommodate the simultaneous embedding of two distinct watermarks, image-level and sequence-level, within each image.

The goal of the optimization process is to determine the optimum strength factor and embedding positions that maximize the robustness while ensuring imperceptibility. We use PSNR to measure the imperceptibility and NC to evaluate the robustness. The NC value is calculated using Equation 11, and the PSNR value is calculated as follows [42]:

$$PSNR = 10log_{10}\left(\frac{255^2}{(\frac{1}{K\times L})\sum_{i=1}^{K}\sum_{j=1}^{L}(Im_{i,j}^{o}-Im_{i,j}^{w})^2}\right) \tag{15}$$

where, $K\times L$ is the size, $Im_{i,j}^{o}$ and $Im_{i,j}^{w}$ are pixels of original and watermarked images.

During each generation, the image-level and sequence-level watermarks are embedded into a sequence of texture images. The PSNR of each watermarked image is calculated to assess the imperceptibility. The solution is penalized if any PSNR value falls below a predefined threshold of 40 dB. Past studies have shown that a PSNR value of 40 dB and above is necessary for high-quality images with visually imperceptible watermarks [47]. Otherwise, a series of attacks is applied to the watermarked images. Finally, the robustness is evaluated by extracting the watermarks from the attacked images and calculating the NC value. This cycle repeats until a predetermined condition is met, yielding the best solution. Figure 7 shows the schematic diagram of the optimization process.

The basic steps of the fitness value calculation of each solution for texture image watermarking are given as follows:

Input: A candidate solution, $x_i = sf, p_1, p_2, ..., p_8$, generated by the ABC algorithm.
Output: Average NC value, NC_{avg}, of candidate solution, x_i .

Step 1: Embed the image-level, W_j^i, and sequence-level, W^s , watermarks into all central images, CIm_j^o, of textures images, TIm_j^o, with the embedding parameters obtained from solution, x_i.

Step 2: Calculate the PSNR values, $PSNR_j$, of all watermarked central images, CIm_j^w.

Step 3: If any of the PSNR values, $PSNR_j$, are lower than the PSNR threshold, Q_{th}, then penalize the solution and return ϵ (a small positive number) to the ABC algorithm as fitness value.

Step 4: Apply n attacks to whole watermarked texture image, TIm_j^w, and obtain the n attacked images, $TIm_{j,n}^A$.

Step 5: Extract the image-level, ${W'}_{j,n}^{i}$, and sequence-level, ${W'}_{j,n}^{s}$, watermarks from the attacked texture images, $TIm_{j,n}^A$.

Step 6: Calculate the average NC, $NC_{avg,i}$, of all extracted watermarks, ${W'}_{j,n}^{i}$ and ${W'}_{j,n}^{s}$.

Step 7: Apply Steps 4-6 to all watermarked texture images, TIm_j^w.

Step 8: Calculate the average NC, NC_{avg}, of the average NC values obtained from all images.

Step 9: Return average NC, NC_{avg}, to the ABC algorithm as fitness value.

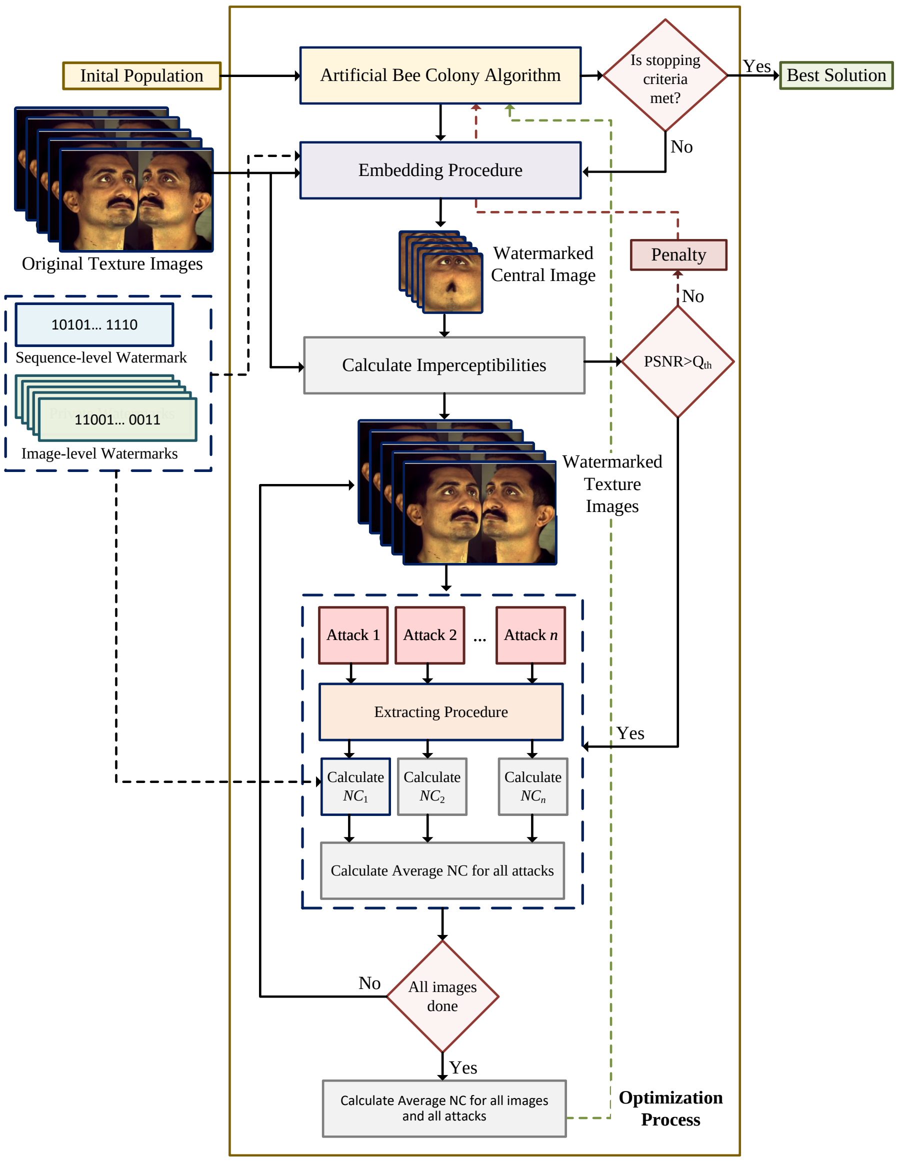


Fig. 7: Optimization process of texture image watermarking

5 Experimental results

This section presents the experimental results for 3D mesh sequence and texture image watermarking. The experiments were conducted in MATLAB R2023b on a laptop with an i7-12700H and 16GB RAM. For 3D mesh sequences, we evaluated the method using different watermark sizes and tested robustness against order change attacks (Section 5.1). For texture images, we optimized the parameters for individual and batch watermarking, assessed robustness, and analyzed performance against order changes (Section 5.2).

5.1 Experimental results of 3D mesh sequence watermarking

This section presents the experimental results for the proposed 3D mesh watermarking method. We tested its performance using various watermark configurations: 16-bit mesh-level with 16-bit sequence-level, 16-bit mesh-level with 32-bit sequence-level, and 32-bit mesh-level with 32-bit sequence-level watermarks. The experiments used sequential 3D meshes from 4D Turkish language data captured by the 3dMD system. A sample size of five meshes (Mesh 1 through Mesh 5) was selected for determining the strength factor and conducting the tests, constrained by time availability.

In the experiments, to evaluate robustness, we subjected the watermarking method to five distinct attacks: quantization (level 9), smoothing (Laplacian filter with a scale factor of 0.05 over 10 iterations), subdivision (midpoint-split), noise addition (level 0.1), and simplification (reduction factor 0.95). These attacks are used during the optimization process to determine the optimal strength factor. Additionally, we tested against content-preserving attacks, such as translation, rotation, scaling, and vertex reordering.

The ABC algorithm was employed to determine the optimal strength factor for the mesh sequence. Performance was assessed through 30 repetitions of experiments for different watermark sizes. Table 2 reports the best strength factors from these runs. The 32-bit mesh-level and 16-bit sequence-level configuration were not tested, as the mesh-level watermark primarily defends against reordering attacks. The sequence-level watermark requires more bits for effective primary copyright protection.

5.1.1 Experiments for 32-bit mesh-level and 32-bit sequence-level watermarks

This subsection presents the results of the experiments using 32-bit mesh-level and 32-bit sequence-level watermarks. The optimum strength factor, 0.1279, was used in the experiments, which were obtained from the optimization process. The meshes were watermarked using the optimum strength factor, and an imperceptibility analysis was then performed. For imperceptibility analysis, we used the Vertex Signal-to-Noise Ratio (VSNR) [27], Signal-to-noise ratio (SNR) [48], Root Mean Square Error (RMSE) [49], Hausdorff distance (HD) [43], Average move distance (AvgD) [48], and the maximum movement distance (MaxD) [48] metrics in addition to the PSNR. Table 3 shows the result of these metrics. As can be seen from the table, the PSNR values

Table 2: Optimum strength factor obtained from the optimization process.

Watermarks sizes	Strength factor
32-bit mesh-level and 32-bit sequence-level	0.1279
16-bit mesh-level and 32-bit sequence-level	0.0633
16-bit mesh-level and 16-bit sequence-level	0.0369

Table 3: Imperceptibility results for 32-bit mesh-level 32-bit sequence-level watermarks.

Metric	Mesh 1	Mesh 2	Mesh 3	Mesh 4	Mesh 5	Average
PSNR	70.5411	70.3883	70.9304	69.9923	70.2194	70.4143
VSNR	116.9580	115.9991	117.0170	115.6918	115.8319	116.2996
SNR	58.4790	57.9995	58.5085	57.8459	57.9159	58.1498
RMSE	0.1008	0.1073	0.1012	0.1082	0.1075	0.1050
HD	0.3261	0.3086	0.39012	0.3809	0.3498	0.3511
AvgD	0.0588	0.0617	0.0580	0.0617	0.0614	0.0603
MaxD	0.3261	0.3086	0.3901	0.3809	0.3498	0.3511

Table 4: NC results for 32-bit mesh-level 32-bit sequence-level watermarks

	Mesh 1			Mesh 2			Mesh 3			Mesh 4			Mesh 5		
Attacks	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.
Translate	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rotation	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Scaling	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Vertex Reordering	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Quantization	1.0000	0.8140	0.9070	0.9258	0.8767	0.9012	1.0000	0.8488	0.9244	1,0000	0.7514	0.8757	1.0000	0.8072	0.9036
Smoothing	0.8164	0.9718	0.8941	0.8807	0.9459	0.9133	0.9074	1.0000	0.9537	0.8017	1.0000	0.9008	0.9474	0.9411	0.9442
Subdivision	0.5217	0.6262	0.5739	0.8660	0.3889	0.6274	0.7669	0.4763	0.6216	0.7905	0.5381	0.6643	0.6069	0.6669	0.6369
Noise	1.0000	0.9718	0.9859	0.9258	1.0000	0.9629	1.0000	0.9459	0.9729	1.0000	0.9718	0.9859	1.0000	0.9718	0.9859
Simplification	0.7106	0.6120	0.6613	0.4900	0.6063	0.5482	0.6120	0.5145	0.5632	0.5455	0.3500	0.4478	0.6315	0.5882	0.6099
Average	0.8943	**0.8884**	0.8914	0.8987	0.8686	0.8837	**0.9207**	0.8651	0.8929	0.9042	0.8457	0.8749	0.9095	0.8861	**0.8978**

of all watermarked meshes greatly exceeded the target PSNR value of 68. The average VSNR, SNR, RMSE, HD, AvgD, and MaxD values are 116.2996, 58.1498, 0.1050, 0.3511, 0.0603, and 0.3511, respectively.

We subjected the watermarked meshes to nine attacks to robustly demonstrate the effectiveness of the proposed method. In these experiments, 32-bit mesh-level and 32-bit sequence-level watermarks were used. Table 4 shows the NC values of the extracted watermarks obtained from attacked meshes, providing a clear picture of the robustness of our method. The table indicates that our method is robust against various attacks, with an average NC of 0.8914, 0.8837, 0.8929, 0.8749, and 0.8978 obtained against all the attacks of five different meshes. This indicates that our method consistently produces similar robustness results for different meshes. Our method is specifically robust against translation, rotation, scaling, and vertex reordering attacks, as the watermark is obtained without any change. The table also shows that our method is robust against smoothing and noise addition attacks, with the lowest average NC value obtained being 0.8941. However, the proposed method is less robust to subdivision and simplification attacks due to their impact on vertex distribution within bins.

5.1.2 Experiments for 16-bit mesh-level and 32-bit sequence-level watermarks

This section presents the findings of experiments conducted using 16-bit mesh-level and 32-bit sequence-level watermarks. The strength factor used was 0.0633, which was obtained through the optimization process. Table 5 demonstrates the imperceptibility results. As per the table, PSNR values for all watermark mesh are higher than the target PSNR value of 68. The average PSNR result is 70.2431. The average values for

Table 5: Imperceptibility results for 16-bit mesh-level and 32-bit sequence-level watermarks

Metric	Mesh 1	Mesh 2	Mesh 3	Mesh 4	Mesh 5	Average
PSNR	70.0308	70.9502	69.2289	70.1962	70.8092	70.2431
VSNR	115.9372	117.1230	113.6141	116.0995	117.0114	115.9570
SNR	57.9686	58.5615	56.8070	58.0497	58.5057	57.9785
RMSE	0.1070	0.1006	0.1231	0.1057	0.1004	0.1074
HD	0.3513	0.3765	0.3888	0.3468	0.3767	0.3680
AvgD	0.0577	0.0550	0.0660	0.0577	0.0548	0.0582
MaxD	0.3513	0.3765	0.3888	0.3468	0.3767	0.3680

Table 6: NC results for 16-bit mesh-level and 32-bit sequence-level watermarks

	Mesh 1			Mesh 2			Mesh 3			Mesh 4			Mesh 5		
Attacks	M-level	S-level	Avg	M-level	S-level	Avg	M-level	S-level	Avg	M-level	S-level	Avg	M-level	S-level	Avg
Translate	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rotation	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Scaling	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Vertex Reordering	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Quantization	1.0000	0.7778	0.8889	1.0000	0.7514	0.8757	1.0000	0.7882	0.8941	1.0000	0.6581	0.8290	1.0000	0.7276	0.8638
Smoothing	1.0000	0.9219	0.9609	0.9128	0.9459	0.9293	1.0000	1.0000	1.0000	0.9354	0.9459	0.9406	0.9258	0.8677	0.8967
Subdivision	1.0000	0.6351	0.8175	1.0000	0.4537	0.7268	0.8770	0.2690	0.5730	1.0000	0.5185	0.7592	1.0000	0.6262	0.8131
Noise	1.0000	0.8997	0.9498	1.0000	0.8488	0.9244	1.0000	0.9459	0.9729	1.0000	0.9459	0.9729	1.0000	0.9393	0.9696
Simplification	0.5773	0.7050	0.6411	0.0000	0.6288	0.3144	0.4803	0.6135	0.5469	0.0000	0.5882	0.2941	0.5345	0.7050	0.6197
Average	**0.9530**	**0.8822**	**0.9176**	0.8792	0.8476	0.8634	0.9286	0.8463	0.8874	0.8817	0.8507	0.8662	0.9400	0.8740	0.9070

VSNR, SNR, RMSE, HD, AvgD, and MaxD are 115.9570, 57.9785, 0.1074, 0.3680, 0.0582, and 0,3680 0.3680, respectively.

Table 6 presents the NC results of the extracted watermark for a 16-bit mesh-level and 32-bit sequence-level watermark. The average NC values obtained for five different meshes are 0.9176, 0.8634, 0.8874, 0.8662, and 0.9070, respectively. These values indicate that the proposed method is robust against attacks like quantization, smoothing, and noise addition. It is also fully robust against translation, rotation, scaling, and vertex reordering. However, it is less robust against subdivision and simplification attacks.

5.1.3 Experiments for 16-bit mesh-level and 16-bit sequence-level watermarks

This section presents the results of experiments conducted using 16-bit mesh-level and 16-bit sequence-level watermarks. The strength factor parameter of the proposed method was selected as 0.0369, which was determined through the optimization process. Table 7 shows the imperceptibility results in terms of different metrics. The table indicates that the PSNR values for all watermarked meshes are greater than the target PSNR value of 68. The average PSNR result is 73.9579, while the average values for VSNR, SNR, RMSE, HD, AvgD, and MaxD are 123.3867, 61.6933, 0.0701, 0.2454, 0.0395, and 0.2454, respectively. These results indicate that the image quality is of high fidelity. This is the first study on 4D watermarking, specifically focusing on temporal sequence watermarking. Previous studies suggest that a PSNR value of 68 or higher indicates good visual quality for single watermarked meshes. Our results show that the proposed method achieves high visual quality and robustness.

Table 7: Imperceptibility results for 16-bit mesh- and 16-bit sequence-level watermarks

Metric	Mesh 1	Mesh 2	Mesh 3	Mesh 4	Mesh 5	Average
PSNR	73.2514	74.9104	72.8435	74.2100	74.5740	73.9579
VSNR	122.3786	125.0435	120.8432	124.1272	124.5411	123.3867
SNR	61.1893	62.5217	60.4216	62.0636	62.2705	61.6933
RMSE	0.0738	0.0638	0.0812	0.0666	0.0651	0.0701
HD	0.2331	0.2524	0.2577	0.2343	0.2497	0.2454
AvgD	0.0402	0.0362	0.0463	0.0386	0.0364	0.0395
Maxd	0.2331	0.2524	0.2577	0.2343	0.2497	0.2454

Table 8: NC results for 16-bit mesh-level and 16-bit sequence-level watermarks

	Mesh 1			Mesh 2			Mesh 3			Mesh 4			Mesh 5		
Attacks	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.
Translate	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rotation	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Scaling	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Vertex Reordering	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Quantization	1.0000	0.9045	0.9522	1.0000	0.9045	0.9522	1.0000	0.9534	0.9767	1.0000	0.8528	0.9264	1.0000	0.9045	0.9522
Smoothing	1.0000	0.9534	0.9767	0.9128	1.0000	0.9564	0.9198	0.9574	0.9386	0.9354	1.0000	0.9677	0.9258	1.0000	0.9629
Subdivision	1.0000	0.6396	0.8198	1.0000	0.3692	0.6846	0.7844	0.6837	0.7341	1.0000	0.6154	0.8077	0.9258	0.7385	0.8321
Noise	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9534	0.9767	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Simplification	0.6123	0.6396	0.6259	0.4082	0.6030	0.5056	0.3721	0.6963	0.5342	0.0000	0.6963	0.3481	0.7142	0.3481	0.5312
Average	**0.9569**	0.9041	**0.9305**	0.9246	0.8752	0.8999	0.8974	**0.9160**	0.9067	0.8817	0.9072	0.8944	0.9518	0.8879	0.9198

The results of an experiment conducted using 16-bit mesh-level and 16-bit sequence-level watermarks applying different types of attacks are displayed in Table 8. The average NC values obtained from Mesh 1 to Mesh 5 are 0.9305, 0.8999, 0.9067, 0.8944, and 0.9198, respectively. These results clearly indicate that our method consistently produces similar robustness results for different meshes. The table shows that the proposed method is robust against a wide range of attacks, including quantization, smoothing, noise addition, translation, rotation, scaling, and vertex reordering. However, it can be said that the method is less robust to subdivision and simplification attacks.

5.1.4 Experiments for changing the order of the meshes in 3D mesh sequence

In the proposed method, mesh-level watermarks are created based on the order of the meshes in the 3D mesh sequence. If the order of the meshes is altered, the extracted watermark and the newly generated mesh-level watermark will not match. This helps in detecting tampering with the order of meshes in the 3D mesh sequence. We conducted a test on the sequence order changing attacks on the watermarked meshes sequence, which are watermarked 16-bit mesh-level and 16-bit sequence-level watermarks. The NC results of the extracted watermarks from the order-changing attack applied to watermarked meshes are presented in Table 9.

According to Table 9, the NC values of the mesh-level and sequence-level watermarks extracted from Mesh 2 are both 1.000, indicating that the order of Mesh 2 is correct. However, for other meshes, the NC values of the extracted sequence-level watermarks are 1.000, while the NC values of the mesh-level watermarks are lower than 0.65. This shows that the orders of Mesh 1, Mesh 3, Mesh 4, and Mesh 5 have been

Table 9: NC values of the extracted watermarks from watermarked 3D mesh sequence against order change attack

Mesh	Original Order	Changed Order	M-level	S-level
Mesh 1	1	4	0.4330	1.0000
Mesh 2	2	2	1.0000	1.0000
Mesh 3	3	5	0.5241	1.0000
Mesh 4	4	1	0.4330	1.0000
Mesh 5	5	3	0.5241	1.0000

changed by unauthorized individuals. On the other hand, Table 9 also shows that the sequence-level watermark can be obtained without errors, even after the reordering of meshes. This implies that the sequence-level watermark can be used as an important asset for copyright claims in 4D data, irrespective of any modifications to the mesh order.

5.2 Experimental results of the texture image watermarking method

In this section, we present the experiments conducted to evaluate the performance of the proposed texture image watermarking method. Firstly, we compared the performance of the proposed method with Gul and Toprak's method, which is the basis of the proposed method. Then, we evaluated the imperceptibility and robustness of the proposed method by using optimal embedding parameters that were determined for a single image. Finally, we assessed the imperceptibility and robustness of the proposed method by using embedding parameters that were determined for all images. We conducted these experiments using sequential texture images taken from 4D data samples for the Turkish language generated using the 3dMD system. We used the first image (Image 1) taken from the sequential images in the 4D data for the experiments conducted using a single image. In the experiments planned to be conducted for all images, we used Image 1, Image 2, Image 3, Image 4, and Image 5 images taken from the sequential images in 4D data. Figure 8 shows the original texture images used in the experiments.

5.2.1 Comparison of the proposed image watermarking method with the baseline method

This section presents a comparison between the proposed image watermarking technique and Gul and Toprak's method. To ensure a fair comparison, we used the same image and selected the same region of the image for watermark embedding. We used the 512×512 center image from Image 1 as input to the embedding process in the optimization process of Gul and Toprak's method. The attacks in the optimization process were applied to the entire watermarked Image 1. Following the optimization process, we obtained the optimum strength factor and embedding positions for "Image 1" using Gul and Toprak's method. To compare the proposed method with Gul and Toprak's method, we performed experiments on the same image, Image 1,

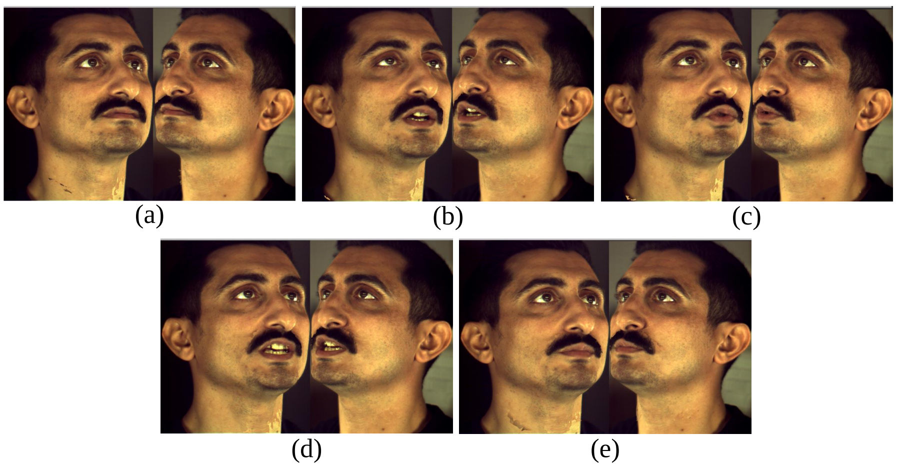


Fig. 8: Original texture images: (a) Image 1, (b) Image 2, (c) Image 3, (d) Image 4, and (e) Image 5

Table 10: Optimum parameters of best solutions obtained from the optimization process of proposed and Gul and Toprak's methods.

Method	Strength factor	Embedding positions							
Gul and Toprak	26.5085	7	13	8	15	19	9	14	29
Proposed	59.1420	8	29	11	39	23	4	7	9

that was used for Gul and Toprak's method. We repeated the optimization process of both methods 30 times and obtained the best solutions. Table 10 shows the optimum embedding parameters of the best solutions obtained for both methods.

Table 11 provides a clear comparison of the PSNR results of both methods using their optimum parameters. It is evident from the table that both methods exhibit good imperceptibility, thanks to their quality thresholds. Notably, Gul and Toprak's method achieves a slightly higher PSNR result than the proposed method. However, the difference in performance between the two methods is negligible, indicating their comparable quality. Both methods meet the required quality standards of 40 or higher, which indicates good visual quality for the watermarked images.

In addition to Gul and Toprak's baseline approach, the proposed texture image watermarking method was also compared to Zhou et al.'s method [50]. To ensure a fair comparison, the watermark was embedded in the 512×512 center image from Image 1, following the same procedure as the baseline evaluation. Zhou et al.'s method utilized a quantization step size of 375, which is their recommended size, along with a 64×64 binary KAYU watermark. As shown in Table 11, Zhou et al.'s method achieved a higher PSNR than the other methods.

We applied 13 different attacks to all three methods to compare their robustness. Table 12 shows the NC values (Normalized Correlation, Section 4.1.3) for each method under each attack. The average NC result obtained for the proposed method

Table 11: Imperceptibility comparison of the proposed, Gul and Toprak's, and Zhou et al.'s methods

	Gul and Toprak	Zhou et al.	Proposed
PSNR of whole image	46.9745	**47.4755**	46.9674
PSNR of embedded region	40.0169	**40.5179**	40.0097

Table 12: Robustness comparison of the proposed, Gul and Toprak's, and Zhou et al.'s methods

Attack	Gul and Toprak	Zhou et al.	Proposed
JPEG20	**0.9694**	0.5881	0.9268
JPEG40	**0.9925**	0.7225	0.9768
JPEG60	**0.9982**	0.8365	0.9958
GWN	0.8902	0.8434	**0.9977**
SP	0.9404	0.8794	**1.0000**
MN	0.9663	0.9973	**1.0000**
HE	**0.9994**	0.2367	0.9958
SH	**1.0000**	0.9992	**1.0000**
RO	0.9728	0.9827	**0.9932**
BR	0.9987	0.9957	**1.0000**
MF	0.9994	0.9984	**1.0000**
GF	0.9994	0.9984	**1.0000**
AF	0.9997	0.9988	**1.0000**
Average	0.9790	0.8521	**0.9912**

is higher than the average NC result obtained for Gul and Toprak's method and Zhou et al.'s method. It is evident from the table that the proposed method is more robust against attacks than the others. Conversely, the table shows that Gul and Toprak's method performs better against compression and enhancement attack groups, while the proposed method is more effective against noise, geometric, and filtering attack groups.

5.2.2 Experiments of the proposed method with optimum embedding parameters determined for only one image

This section gives the results of experiments in which we determined the optimum parameters on a single image and then used these optimal parameters to watermark all the images. Table 13 shows the PSNR result of the watermarked images. It can be seen from the table that PSNR values of the watermarked region of Image 2, Image 3, Image 4, and Image 5 images are lower than the predefined quality threshold of 40 dB with the optimum parameters obtained for Image 1. These results indicate that the watermarked images are not of the desired quality. Therefore, it is clear that using the optimum embedding parameters determined using one image for all images in the 4D data is not sufficient.

We have also conducted experiments to evaluate the robustness of the proposed method on all images using the optimum embedding parameters determined for only

Table 13: Imperceptibility results with optimum embedding parameters determined for only one image (PSNR values of the watermarked images)

	Image 1	Image 2	Image 3	Image 4	Image 5
PSNR of whole image	46.9674	46.5407	46.5510	46.5136	46.7001
PSNR of embedded region	40.0097	39.7194	39.7109	39.6530	39.7281

Table 14: Robustness results with optimum embedding parameters determined for only one image (NC values of the extracted watermarks)

	Image 1			Image 2			Image 3			Image 4			Image 5		
Attacks	I-level	S-level	Avg.	I-level	S-level	Avg.	I-level	S-level	Avg.	I-level	S-level	Avg.	I-level	S-level	Avg.
JPEG20	0.8581	0.9955	0.9268	0.7142	0.9868	0.8505	0.8129	0.9823	0.8976	0.6422	0.9866	0.8144	0.7489	0.98214	0.8655
JPEG40	0.9536	1.0000	0.9768	0.9725	1.0000	0.9862	0.9724	1.0000	0.9862	0.9091	1.0000	0.9545	0.9271	1.0000	0.9635
JPEG60	0.9916	1.0000	0.9958	0.9883	1.0000	0.9941	0.9931	1.0000	0.9965	0.9880	1.0000	0.9940	0.9743	1.0000	0.9871
GWN	1.0000	0.9955	0.9977	1.0000	1.0000	1.0000	0.9965	1.0000	0.9982	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
SP	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9957	1.0000	0.9978
MN	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9965	1.0000	0.9982	0.9959	1.0000	0.9979	0.9957	1.0000	0.9978
HE	0.9916	1.0000	0.9958	1.0000	1.0000	1.0000	1.0000	0.9910	0.9955	0.9920	0.9955	0.9938	1.0000	1,0000	1.0000
SH	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
RO	1.0000	0.9865	0.9932	0.9961	0.9955	0.9958	1.0000	1.0000	1.0000	0.9879	1.0000	0.9939	0.9957	0.9865	0.9911
BR	1.0000	1.0000	1.0000	0.9922	0.9865	0.9893	0.9896	0.9955	0.9926	0.9960	0.9955	0.9957	0.9914	0.9955	0.9935
MF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
GF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9965	1.0000	0.9982	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
AF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9965	1.0000	0.9982	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Average	**0.9842**	**0.9982**	**0.9912**	0.9741	0.9976	0.9858	0.9811	0.9976	0.9893	0.9624	**0.9982**	0.9803	0.9714	0.9972	0.9843

one image. Table 14 gives the NC values of the extracted watermarks after applying 13 different attacks to the images. The table shows that Image 1 has the best average NC because it has optimum embedding parameters determined for itself. Nevertheless, the results obtained for all images are satisfactory. However, it is necessary to determine the optimum embedding parameters for all images to achieve the desired imperceptibility.

5.2.3 Experiments with optimum embedding parameters determined for all images

The results presented in Section 5.2.2 reveal that applying optimal embedding parameters derived from a single image to all images in the 4D data results in watermarked images lacking the desired quality, such as low PSNR values. Therefore, our proposed method determines a set of optimum embedding parameters using all images in the sequence. In the experiments, we used five images taken from sequential images in 4D data to demonstrate the performance of the proposed method. The proposed method utilizes the ABC algorithm to determine the optimum embedding parameters. Accordingly, the strength factor was found to be 54.2687, and the embedding positions were determined to be {43 19 11 64 25 8 6 7} in the best solution for the selected five images. We then performed robustness and imperceptibility tests of the proposed method using this best parameter set.

Figure 9 demonstrates the watermarked texture images, and Table 15 shows the PSNR results of the watermarked images using the optimal embedding parameters determined for all images. The imperceptibility results indicate that the PSNR values of all watermarked images are higher than the predetermined threshold value of 40 dB. It is evident from the table that the proposed method maintains the quality of the images above the desired quality level. On the other hand, Tables 13 and 15 reveal that it is necessary to determine the optimal embedding parameters for all

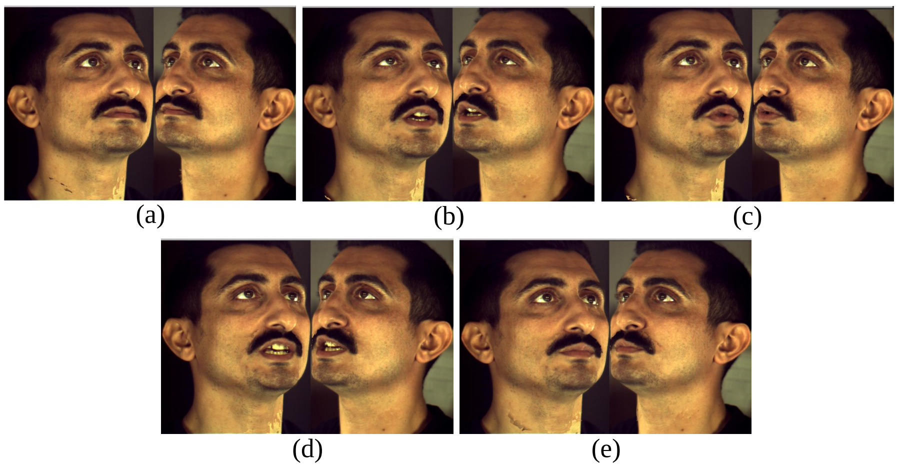


Fig. 9: Watermarked texture images: (a) Image 1 - (e) Image 5

images. As shown in Table 13, experiments conducted with the optimal embedding parameters determined for a single image may result in the watermarked region of the image being below the desired quality limit. However, by determining the optimal embedding parameters for all images using the proposed method, it is observed that the watermarked region of all images is higher than the desired threshold value.

The robustness evaluation of the proposed method using the optimum embedding parameters determined for all five images is performed with 13 different attacks. Table 16 shows the NC values of the extracted watermarks from attacked images. The highest average NC value was obtained from Image 1 as 0.9889, while the lowest average NC value was obtained from Image 3 as 0.9859. For S-level watermarks, the highest average NC value was obtained as 0.9989 from Image 1, while the lowest value was obtained as 0.9969 from Image 2 and Image 3. For I-level watermarks, the highest average NC value was obtained as 0.9789 from Image 1, while the lowest value was obtained as 0.9749 from Image 3. Overall, considering all the results, the proposed method demonstrated satisfactory robustness.

5.2.4 Experiments for changing the order of the images in images sequence

The proposed method generates image-level watermarks to detect any changes in the order of the texture images within an image sequence. If the order of the images is altered, the extracted image-level watermark and the re-generated image-level watermark will not match. This feature allows us to identify any tampering with the order

Table 15: Imperceptibility results with optimum embedding parameters determined for all images (PSNR values of the watermarked images).

	Image 1	Image 2	Image 3	Image 4	Image 5
PSNR of whole image	47.1992	47.0575	46.8427	46.9128	47.2645
PSNR of embedded region	40.2416	40.2362	40.0026	40.0521	40.2925

Table 16: Robustness results with optimum embedding parameters determined for all images (NC values of the extracted watermarks)

	Image 1			Image 2			Image 3			Image 4			Image 5		
Attacks	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg
JPEG20	0.7836	0.9911	0.8874	0.8096	0.9866	0.8981	0.7551	0.9780	0.8665	0.7840	0.9911	0.8876	0.8062	0.9821	0.8942
JPEG40	0.9515	1.0000	0.9757	0.9333	1.0000	0.9666	0.9465	0.9955	0.9710	0.9322	1.0000	0.9661	0.9373	0.9955	0.9664
JPEG60	0.9958	1.0000	0.9979	0.9735	1.0000	0.9867	0.9897	1.0000	0.9948	0.9960	1.0000	0.9980	0.9745	1.0000	0.9872
GWN	1.0000	1.0000	1.0000	0.9961	1.0000	0.9980	0.9965	0.9955	0.9960	1.0000	1.0000	1.0000	1.0000	0.9955	0.9977
SP	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9965	1.0000	0.9982	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
MN	1.0000	1.0000	1.0000	0.9961	1.0000	0.9980	0.9931	1.0000	0.9965	0.9959	0.9955	0.9957	0.9957	1.0000	0.9978
HE	1.0000	1.0000	1.0000	0.9961	0.9955	0.9958	1.0000	1.0000	1.0000	1.0000	0.9955	0.9977	1.0000	1.0000	1.0000
SH	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
RO	0.9958	0.9955	0.9957	1.0000	0.9866	0.9933	1.0000	1.0000	1.0000	0.9959	1.0000	0.9979	1.0000	0.9955	0.9977
BR	1.0000	1.0000	1.0000	0.9961	0.9910	0.9936	0.9965	0.9911	0.9938	1.0000	0.9955	0.9977	0.9915	1.0000	0.9957
MF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
GF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
AF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Average	**0.9789**	**0.9989**	**0.9889**	0.9770	0.9969	0.9869	0.9749	0.9969	0.9859	0.9772	0.9982	0.9877	0.9773	0.9975	0.9874

Table 17: NC values of the extracted watermarks against order change attack

Image	Original order	Changed order	I-level	S-level
Image 1	1	4	0.4735	1.0000
Image 2	2	2	1.0000	1.0000
Image 3	3	5	0.5409	1.0000
Image 4	4	1	0.4735	1.0000
Image 5	5	3	0.5409	1.0000

of the images in the dataset. To further validate the effectiveness of our method, we subjected the watermarked image sequence to order-changing attacks. The results of these attacks are presented in Table 17, providing a clear indication of the method's robustness.

It is seen from Table 17 that the NC values of the I-level and S-level watermarks extracted from Image 2 are 1.000 and the same. This shows that the order of Image 2 is correct. However, for other images, the NC values of the S-level general watermarks are 1.000, while the NC values of the I-level watermarks are lower than 0.65. This demonstrates that the orders of Image 1, Image 3, Image 4, and Image 5 have been changed.

5.3 Demonstration of watermarked 4D face data

This section conducts a subjective analysis of the quality of the watermarked data, in addition to the objective imperceptibility analysis performed for 3D meshes and texture images in the previous sections. Figure 10 shows the original and watermarked 3D mesh sequence with corresponding texture images. It is seen from the figure that the imperceptibility of the watermarked 4D face data is satisfactory. It is also clear that the differences between the original and watermarked data are difficult to detect with the human eye.

5.4 Quality threshold analysis

In this section, a quality threshold analysis is conducted to evaluate the impact of varying threshold values on the performance of the proposed watermarking method,

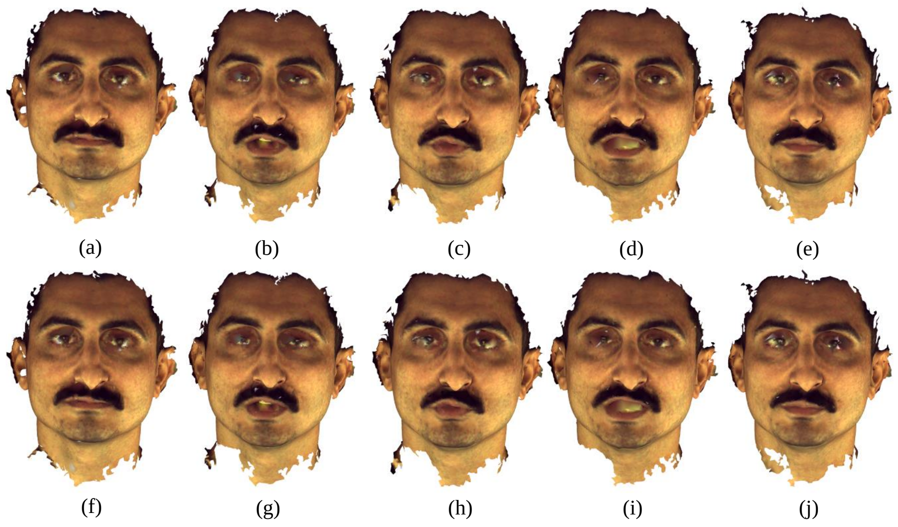


Fig. 10: Original and watermarked 3D mesh sequence with corresponding texture images; Original: (a) Frame 1, (b) Frame 2, (c) Frame 3, (d) Frame 4, and (e) Frame 5; Watermarked: (f) Frame 1, (g) Frame 2, (h) Frame 3, (i) Frame 4, and (j) Frame 5

with respect to both imperceptibility and robustness. Specifically, the objective is to determine how the different quality threshold levels influence the watermark embedding process and the resultant trade-off between visual imperceptibility and robustness to attacks.

The proposed optimization-based watermarking scheme includes a tunable PSNR threshold as a parameter, which guides the ABC algorithm in determining the optimal watermark embedding strength factor. By adjusting this threshold, we observe its effect on two key performance metrics: PSNR, which indicates visual imperceptibility, and NC, which measures the robustness of the watermark. In the experiments, we used the same set of images and mesh sequences as in the previous experiments.

Table 18 illustrates the impact of varying PSNR thresholds on the imperceptibility and robustness of watermarking in 3D mesh sequences. As shown in the table, increasing the quality threshold leads to a decrease in the embedding strength factor. Since the embedding strength directly influences the robustness of the watermarking process, a reduction in this factor also leads to lower average NC values. On the other hand, as expected, the average PSNR values consistently exceed the specified quality thresholds, indicating good visual quality. Previous studies have demonstrated that a PSNR value of 68 or higher corresponds to good visual quality for watermarked 3D meshes [43, 45]. Therefore, in this study, we selected a default quality threshold of 68. However, this threshold can be adjusted according to the specific requirements of the application area.

Table 19 displays the results of the quality threshold analysis for texture image watermarking. A trend similar to that seen in Table 18 can be observed in the results. As the quality threshold increases, the embedding strength factor decreases, which

Table 18: Quality threshold analysis for 3D mesh sequence watermarking

Quality threshold	Strength factor	Average NC	Average PSNR
58 dB	0.2627	0.8517	58.9120 dB
68 dB	0.0697	0.8416	68.8501 dB
78 dB	0.0126	0.7675	80.0070 dB

Table 19: Quality threshold analysis for texture images watermarking

Quality threshold	Strength factor	Average NC	Average PSNR
35 dB	99.4603	0.9938	35.2042 dB
40 dB	54.2687	0.9761	40.1650 dB
45 dB	31.4892	0.9526	45.2587 dB

leads to lower NC values. This illustrates the inherent trade-off between imperceptibility and robustness: while a higher quality threshold enhances visual quality, it may simultaneously reduce the robustness of the watermarking process. Previous studies have indicated that a Peak Signal-to-Noise Ratio (PSNR) of 40 dB or higher is necessary to achieve high-quality images with visually imperceptible watermarks [47]. Consequently, for this study, we have chosen 40 dB as the quality threshold for texture image watermarking. However, users have the flexibility to adjust this value based on their specific requirements. By modifying the quality threshold, they can identify the embedding strength factor that offers optimal robustness for their chosen level of visual quality.

5.5 Experiments on 4D Welsh dataset

To validate the effectiveness and generalizability of the proposed approach on other facial data captured by the 3dMD system, we conducted experiments using a subset of the CymruFluency dataset [51]. The CymruFluency dataset consists of recordings from 33 speakers, each pronouncing ten Welsh phrases, selected by a Welsh language expert according to increasing linguistic difficulty.

For our evaluation, we used the recordings of Participant 4 saying "Eisteddfod yr Urdd" (the first phrase). A representative sample consisting of five sequential 3D meshes (NMesh 1 to NMesh 5) and their corresponding texture images (NImage 1 to NImage 5) from the 4D Welsh dataset was selected. These samples were used to determine optimal watermark embedding parameters and thoroughly assess the robustness and visual imperceptibility of the proposed method on additional data generated with the 3dMD system.

The optimal strength factor parameter for the proposed 3D mesh sequence watermarking method, obtained via the optimization process, was determined to be 0.0401. Table 20 presents imperceptibility results across various quality metrics. It can be seen that the PSNR values for all watermarked meshes consistently exceed the predefined visual quality threshold of 68 dB, achieving an average PSNR value of 75.0506 dB.

Additionally, the obtained averages for VSNR, SNR, RMSE, HD, AvgD, and MaxD were 124.9961, 62.4980, 0.0661, 0.2387, 0.0344, and 0.2387, respectively. Collectively, these metrics confirm that the visual quality of watermarked meshes is preserved at a high quality.

Table 20: Imperceptibility results for 3D mesh sequence of 4D Welsh data

Metric	NMesh 1	NMesh 2	NMesh 3	NMesh 4	NMesh 5	Average
PSNR	75.6465	75.8613	75.1101	77.3966	71.2387	75.0506
VSNR	126.1048	126.9385	125.1580	129.2786	117.5004	124.9961
SNR	63.0524	63.4692	62.5790	64.6393	58.7502	62.4980
RMSE	0.0603	0.0574	0.0648	0.0503	0.0979	0.0661
HD	0.2067	0.2133	0.2235	0.1943	0.3556	0.2387
AvgD	0.0313	0.0289	0.0327	0.0262	0.0530	0.0344
Maxd	0.2067	0.2133	0.2235	0.1943	0.3556	0.2387

The robustness evaluation results of the 3D mesh sequence watermarking, presented in Table 21. As shown in the table, the proposed approach demonstrates a strong robustness against common attacks, including quantization, smoothing, and noise addition. Moreover, the proposed method is seen to be completely robust against geometric transformations and vertex reordering attacks, such as translation, rotation, and scaling. Specifically, the average NC values for the five meshes ranged from 0.8962 to 0.9414, confirming consistently high robustness.

Table 21: NC results for 3D mesh sequence of 4D Welsh data

	NMesh 1			NMesh 2			NMesh 3			NMesh 4			NMesh 5		
Attacks	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.	M-level	S-level	Avg.
Translate	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Rotation	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Scaling	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Vertex Reordering	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Quantization	1.0000	0.9534	0.9767	1.0000	0.9045	0.9522	1.0000	1.0000	1.0000	0.9428	1.0000	0.9714	1.0000	0.7385	0.8692
Smoothing	0.9128	1.0000	0.9564	0.9128	1.0000	0.9564	0.9198	1.0000	0.9599	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Subdivision	0.7071	0.6674	0.6872	0.6172	0.4045	0.5108	0.4803	0.5330	0.5066	0.6324	0.6674	0.6499	1.0000	0.6963	0.8481
Noise	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Simplification	0.5163	0.6030	0.5597	0.6804	0.6154	0.6479	0.4529	0.7462	0.5995	0.5590	0.7462	0.6526	0.7715	0.7385	0.7550
Average	0.9040	0.9138	0.9089	0.9123	0.8805	0.8964	0.8726	0.9199	0.8962	0.9038	**0.9348**	0.9193	**0.9746**	0.9081	**0.9414**

For the texture image sequence, the optimization process identified an optimal strength factor of 57.1624, along with embedding positions at indices {5, 13, 22, 19, 6, 16, 43, 2}. Imperceptibility and robustness evaluations were then performed using these optimized parameters. As indicated in Table 22, all PSNR values for the watermarked texture images significantly exceeded the targeted threshold of 40 dB, clearly demonstrating high visual quality. Furthermore, the robustness results provided in Table 23 indicate that the method maintained excellent robustness to attacks, with average NC values ranging from 0.9840 to 0.9902, confirming strong robustness for texture images.

Finally, Figure 11 shows the original and watermarked 3D mesh sequences along with their texture images. These results demonstrate the application of the proposed method on a sample from the 4D Welsh dataset captured by the 3dMD system. From

Table 22: PSNR results of the watermarked texture image sequence of 4D Welsh data

	NImage 1	NImage 2	NImage 3	NImage 4	NImage 5
PSNR of whole image	47.5833	47.2715	47.3423	47.4770	47.1478
PSNR of embedded region	40.4594	40.3476	40.3297	40.3939	40.1100

Table 23: NC results for texture image sequence of 4D Welsh data

	NImage 1			NImage 2			NImage 3			NImage 4			NImage 5		
Attacks	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg	I-level	S-level	Avg
JPEG20	0.8491	0.9728	0.9110	0.7876	0.9732	0.8804	0.8419	0.9686	0.9053	0.7756	0.9821	0.8789	0.7689	0.9820	0.8754
JPEG40	0.9421	1.0000	0.9710	0.9134	0.9865	0.9500	0.9440	1.0000	0.9720	0.9393	0.9910	0.9651	0.8988	0.9955	0.9471
JPEG60	0.9916	1.0000	0.9958	0.9844	1.0000	0.9922	0.9896	1.0000	0.9948	0.9880	1.0000	0.9940	0.9700	0.9955	0.9827
GWN	1.0000	0.9911	0.9955	1.0000	1.0000	1.0000	1.0000	0.9868	0.9934	0.9920	0.9732	0.9826	0.9916	0.9955	0.9935
SP	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9955	0.9977
MN	1.0000	1.0000	1.0000	0.9922	1.0000	0.9961	0.9965	1.0000	0.9982	0.9960	1.0000	0.9980	1.0000	1.0000	1.0000
HE	1.0000	1.0000	1.0000	1.0000	0.9955	0.9977	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
SH	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
RO	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9959	0.9955	0.9957	1.0000	1.0000	1.0000
BR	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	0.9955	0.9977	0.9960	1.0000	0.9980	0.9957	0.9955	0.9956
MF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
GF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
AF	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Average	**0.9833**	**0.9972**	**0.9902**	0.9752	0.9965	0.9858	0.9824	0.9962	0.9893	0.9756	0.9955	0.9855	0.9711	0.9969	0.9840

the figure, it is clear that the imperceptibility of the watermarked 4D face data is very good. Additionally, the differences between the original and watermarked data are almost undetectable to the naked eye.

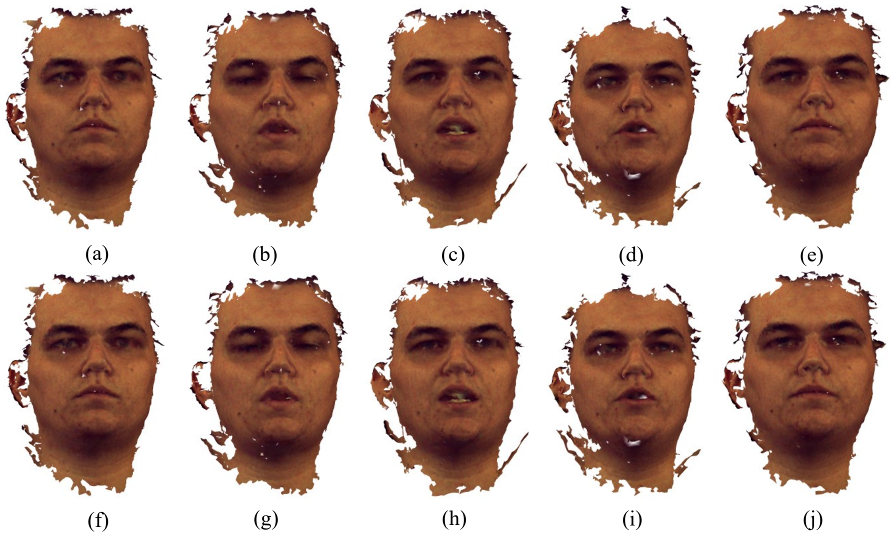


Fig. 11: Original and watermarked 3D mesh sequence with corresponding texture images from 4D Welsh Data; Original: (a) Frame 1, (b) Frame 2, (c) Frame 3, (d) Frame 4, and (e) Frame 5; Watermarked: (f) Frame 1, (g) Frame 2, (h) Frame 3, (i) Frame 4, and (j) Frame 5

6 Conclusion

This work introduces a novel 4D watermarking technique for the copyright protection of 4D face data. To the best of our knowledge, this is the first method specifically developed for 4D watermarking. Unlike most existing watermarking techniques that rely on manually set embedding parameters, our approach employs the Artificial Bee Colony (ABC) algorithm to optimize these parameters. Furthermore, the proposed 3D mesh watermarking method is the first to ensure the desired imperceptibility of watermarked meshes.

As a pioneering effort in the 4D watermarking domain, we have also created a unique 4D face dataset specifically for this purpose. Experimental results validate the robustness of our method, demonstrating its resilience against a variety of attacks while maintaining high quality in the watermarked cover data, thanks to the quality threshold. Importantly, our method can detect any manipulation of the mesh or texture image sequence order, ensuring reliable security for 4D face data.

However, the method has limitations, including relatively low capacity and slightly reduced robustness against subdivision and simplification attacks for 3D mesh sequence watermarking. Future work will focus on overcoming our limitations and extending the application of our method to various types of 4D data with different structures, including audio data. Additionally, we aim to enhance the capacity and robustness of the 3D mesh sequence watermarking approach, particularly improving its resistance to subdivision and simplification attacks, as well as explore alternative techniques for 4D watermarking.

Acknowledgements. This study received support from the Scientific and Technical Research Council of Turkey (TUBITAK) under the 2219 postdoctoral fellowship support program, project number 1059B192202811. Tam acknowledges partial support from a Royal Society fund (IEC/NSFC/211159). Additionally, the study received partial visiting researcher support from the Faculty of Engineering at Swansea University. The authors express gratitude to TUBITAK, Kayseri University, and Swansea University for their support. For the purpose of Open Access, the author has applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript (AAM) version arising from this submission.

References

[1] Farri, E., Ayubi, P.: A robust digital video watermarking based on ct-svd domain and chaotic dna sequences for copyright protection. Journal of Ambient Intelligence and Humanized Computing **14**(10), 13113–13137 (2023)

[2] Yamni, M., Karmouni, H., Sayyouri, M., Qjidaa, H.: Robust audio watermarking scheme based on fractional charlier moment transform and dual tree complex wavelet transform. Expert Systems with Applications **203**, 117325 (2022)

[3] Su, Q., Chen, S., Wang, H., Cao, H., Hu, F.: An efficient watermarking scheme for dual color image with high security in 5g environment. Expert Systems with Applications, 123818 (2024)

[4] Ozkaya, H., Aslantas, V.: A triple self-embedding fragile watermarking scheme for image tamper detection and recovery. IEEE Access (2024)

[5] Zhu, L., Zhao, Y., Fang, Y., Wang, J.: A novel robust digital image watermarking scheme based on attention u-net++ structure. The Visual Computer **40**(12), 8791–8807 (2024)

[6] Moosazadeh, M., Ekbatanifard, G.: A new dct-based robust image watermarking method using teaching-learning-based optimization. Journal of Information Security and Applications **47**, 28–38 (2019)

[7] Soualmi, A., Alti, A., Laouamer, L.: An imperceptible watermarking scheme for medical image tamper detection. International Journal of Information Security and Privacy (IJISP) **16**(1), 1–18 (2022)

[8] Gul, E.: A blind robust color image watermarking method based on discrete wavelet transform and discrete cosine transform using grayscale watermark image. Concurrency and Computation: Practice and Experience **34**(22), 6884 (2022)

[9] Han, S.-H., Chu, C.-H.: Content-based image authentication: current status, issues, and challenges. International Journal of Information Security **9**(1), 19–32 (2010)

[10] Qiu, Y., Jiao, S., Su, Q.: Enhancing color image watermarking via fast quaternion schur decomposition: a high-quality blind approach. The Visual Computer **41**(7), 4497–4515 (2025)

[11] Kumaran, V.S., Manikandan, T., Dhanaraj, R.K., Al-Shehari, T., Alsadhan, N.A., Selvarajan, S.: A secure medical image encryption technique based on dna cryptography with elliptic curves. Scientific Reports **15**(1), 20003 (2025)

[12] Soualmi, A., Laouamer, L., Alti, A.: A novel intelligent approach for color image privacy preservation. Multimedia Tools and Applications **83**(33), 79481–79502 (2024)

[13] Gul, E., Ozturk, S.: A secret image sharing method based on block-wise cheating detection and recovery on shadow images. Computers and Electrical Engineering **110**, 108882 (2023)

[14] Soualmi, A., Laouamer, L., *et al.*: A blind watermarking approach based on hybrid imperialistic competitive algorithm and surf points for color images' authentication. Biomedical Signal Processing and Control **84**, 105007 (2023)

[15] Azizoglu, G., Toprak, A.N.: A novel reversible fragile watermarking method in dwt domain for tamper localization and digital image authentication. Biomedical Signal Processing and Control **84**, 105015 (2023)

[16] Frattolillo, F.: Watermarking protocols: An excursus to motivate a new approach. International Journal of Information Security **17**(5), 587–601 (2018)

[17] Borah, S., Borah, B.: Watermarking techniques for three dimensional (3d) mesh authentication in spatial domain. 3D Research **9**(3), 43 (2018)

[18] 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 **55**(1), 142–155 (2006)

[19] Jallouli, M., Sayahi, I., Mabrouk, A.B.: Robust crypto-watermarking approach based on spherical harmonics and aes algorithm for 3d mesh safe transmission. Multimedia Tools and Applications **81**(27), 38543–38567 (2022)

[20] Singh, P., Devi, K.J.: Blind, secured and robust watermarking for 3-d polygon mesh using vertex curvature. International Journal of Advanced Computer Science and Applications **12**(8) (2021)

[21] Cosker, D., Krumhuber, E., Hilton, A.: A facs valid 3d dynamic action unit database with applications to 3d dynamic morphable facial modeling. In: 2011 International Conference on Computer Vision, pp. 2296–2303 (2011). IEEE

[22] Zhang, X., Yin, L., Cohn, J.F., Canavan, S., Reale, M., Horowitz, A., Liu, P.: A high-resolution spontaneous 3d dynamic facial expression database. In: 2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), pp. 1–6 (2013). IEEE

[23] Lee, J.-S., Liu, C., Chen, Y.-C., Hung, W.-C., Li, B.: Robust 3d mesh zero-watermarking based on spherical coordinate and skewness measurement. Multimedia Tools and Applications **80**(17), 25757–25772 (2021)

[24] Narendra, M., Valarmathi, M., Anbarasi, L.J.: Watermarking techniques for three-dimensional (3d) mesh models: a survey. Multimedia systems **28**(2), 623–641 (2022)

[25] Medimegh, N., Belaid, S., Atri, M., Werghi, N.: 3d mesh watermarking using salient points. Multimedia Tools and Applications **77**, 32287–32309 (2018)

[26] Delmotte, A., Tanaka, K., Kubo, H., Funatomi, T., Mukaigawa, Y.: Blind 3d-printing watermarking using moment alignment and surface norm distribution. IEEE Transactions on Multimedia **23**, 3467–3482 (2020)

[27] Sharma, N., Panda, J.: Statistical watermarking approach for 3d mesh using local curvature estimation. IET information security **14**(6), 745–753 (2020)

[28] Medimegh, N., Belaid, S., Atri, M., Werghi, N.: Statistical 3d watermarking algorithm using non negative matrix factorization. Multimedia Tools and Applications

79, 25889–25904 (2020)

[29] Choi, H.-Y., Jang, H.-U., Son, J., Lee, H.-K.: Blind 3d mesh watermarking based on cropping-resilient synchronization. Multimedia Tools and Applications **76**, 26695–26721 (2017)

[30] Hu, R., Xie, L., Yu, H., Ding, B., et al.: Applying 3d polygonal mesh watermarking for transmission security protection through sensor networks. Mathematical problems in engineering **2014** (2014)

[31] Kim, M.-S., Prost, R., Chung, H.-Y., Jung, H.-Y.: A blind watermarking for 3-d dynamic mesh model using distribution of temporal wavelet coefficients. In: International Workshop on Multimedia Content Representation, Classification and Security, pp. 257–264 (2006). Springer

[32] Tsai, J.-S., Hsiao, J.-T., Huang, W.-B., Kuo, Y.-H.: Geodesic-based robust blind watermarking method for three-dimensional mesh animation by using mesh segmentation and vertex trajectory. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1757–1760 (2012). IEEE

[33] Mouhamed, M.R., Soliman, M.M., Darwish, A., Hassanien, A.E.: Watermarking 3d printing data based on coyote optimization algorithm. Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges, 603–624 (2021)

[34] Narendra, M., Valarmathi, M., Anbarasi, L.J.: Optimization of 3d triangular mesh watermarking using aco-weber's law. KSII Transactions on Internet and Information Systems (TIIS) **14**(10), 4042–4059 (2020)

[35] Mouhamed, M.R., Soliman, M.M., Darwish, A.A., Hassanien, A.E.: A robust 3d mesh watermarking approach based on genetic algorithm. In: 2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS), pp. 408–413 (2019). IEEE

[36] Tamane, S.C., Deshmukh, R.R., Jadhavpatil, V.: Optimization of blind 3d model watermarking using wavelets and dct. In: 2013 4th International Conference on Intelligent Systems, Modelling and Simulation, pp. 270–275 (2013). IEEE

[37] Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department (2005)

[38] Hou, G., Ou, B., Peng, F., Long, M.: A traitor tracing and access control method for encrypted 3d models based on cp-abe and fair watermark. IEEE Signal Processing Letters (2024)

[39] Alhammad, S.M., Ahmed, N., Abbas, S., Abulkasim, H., Elhadad, A.: Robust 3d

object watermarking scheme using shape features for copyright protection. PeerJ Computer Science **10**, 2020 (2024)

[40] Hu, J., Dai, M., Wang, X., Xie, Q., Zhang, D.: Robust 3d watermarking with high imperceptibility based on emd on surfaces. The Visual Computer **40**(11), 7685–7700 (2024)

[41] Andrews, G., Endean, S., Dyke, R., Lai, Y., Ffrancon, G., Tam, G.K.: Hdfd—a high deformation facial dynamics benchmark for evaluation of non-rigid surface registration and classification. arXiv preprint arXiv:1807.03354 (2018)

[42] Gul, E., Toprak, A.N.: Contourlet and discrete cosine transform based quality guaranteed robust image watermarking method using artificial bee colony algorithm. Expert Systems with Applications **212**, 118730 (2023)

[43] Narendra, M., Valarmathi, M., Anbarasi, L.J., Sarobin, M.V.R., Al-Turjman, F.: High embedding capacity in 3d model using intelligent fuzzy based clustering. Neural Computing and Applications **34**(20), 17783–17792 (2022)

[44] Korsawe, J.: Minimal Bounding Box. https://www.mathworks.com/matlabcentral/fileexchange/18264-minimal-bounding-box, MATLAB Central File Exchange. Retrieved May 13, 2024

[45] Arthy, R., Mala, K., Suresh Babu, R.: Hypothesis-based vertex shift method for embedding secret logos in the geometric features of 3d objects. Turkish Journal of Electrical Engineering and Computer Sciences **27**(6), 4518–4529 (2019)

[46] Tian, C., Wen, R.-H., Zou, W.-P., Gong, L.-H.: Robust and blind watermarking algorithm based on dct and svd in the contourlet domain. Multimedia Tools and Applications **79**, 7515–7541 (2020)

[47] Cheddad, A., Condell, J., Curran, K., Mc Kevitt, P.: Digital image steganography: Survey and analysis of current methods. Signal processing **90**(3), 727–752 (2010)

[48] Peng, F., Long, B., Long, M.: A semi-fragile reversible watermarking for authenticating 3d models based on virtual polygon projection and double modulation strategy. IEEE Transactions on Multimedia **25**, 892–906 (2021)

[49] El Zein, O.M., El Bakrawy, L.M., Ghali, N.I.: A robust 3d mesh watermarking algorithm utilizing fuzzy c-means clustering. Future Computing and Informatics Journal **2**(2), 148–156 (2017)

[50] Zhou, Q., Shen, S., Yu, S., Duan, D., Yuan, Y., Lv, H., Lin, H.: A novel robust image watermarking algorithm based on polar decomposition and image geometric correction. The Visual Computer **40**(5), 3303–3330 (2024)

[51] Bali, A.P.S., Tam, G.K.L., Siris, A., Andrews, G., Lai, Y., Tiddeman, B., Ffrancon, G.: Cymrufluency - a fusion technique and a 4d welsh dataset for welsh fluency analysis. In: Advanced Concepts for Intelligent Vision Systems (ACIVS). Springer, Japan (2025). https://doi.org/10.5281/zenodo.15397513 . https://doi.org/10.5281/zenodo.15397513