






# Blind 3D-Printing Watermarking Using Moment Alignment and Surface Norm Distribution

Arnaud Delmotte , Kenichiro Tanaka , *Member, IEEE*, Hiroyuki Kubo , *Member, IEEE*, Takuya Funatomi , *Member, IEEE*, and Yasuhiro Mukaigawa , *Member, IEEE*

**Abstract**—The recent development of 3D printing technology has brought concerns about its potential misuse, such as in copyright infringement, and crimes. Although there have been many studies on blind 3D mesh watermarking for the copyright protection of digital objects, methods applicable to 3D printed objects are rare. In this paper, we propose a novel blind watermarking algorithm for 3D printed objects with applications for copyright protection, traitor tracing, object identification, and crime investigation. Our method allows us to embed a few bits of data into a 3D-printed object, and retrieve it by 3D scanning without requiring any information about the original mesh. The payload is embedded on the object's surface by slightly modifying the distribution of surface norms, that is, the distance between the surface, and the center of gravity. It is robust to resampling and can work with any 3D printer, and scanner technology. In addition, our method increases the capacity, and resistance by subdividing the mesh into a set of bins, and spreading the data over the entire surface to negate the effect of local printing artifacts. The method's novelties include extending the vertex norm histogram to a continuous surface, and the use of 3D moments to synchronize a watermark signal in a 3D-printing context. In the experiments, our method was evaluated using a public dataset against center, orientation, minimum, and maximum norm misalignments; a printing simulation; and actual print/scan experiments using a standard 3D printer, and scanner.

**Index Terms**—3D moments, 3D printing, 3D scanning, blind watermarking, surface norms.

## I. INTRODUCTION

THREE-DIMENSIONAL (3D) printing has expanded substantially in recent years, with affordable and reliable printers starting at a few hundred dollars, as well as an increasing diversity of supported materials, such as plastic, metal, concrete, ceramic, and even food. The market sales of consumer printers have outpaced industrial printers since 2011 [1], [2]. Similarly,

Manuscript received March 9, 2020; revised June 22, 2020 and August 10, 2020; accepted September 7, 2020. Date of publication September 22, 2020; date of current version October 19, 2021. This work was supported by JSPS KAKEN JP17K19979. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaochun Cao. (*Corresponding author: arnaud.delmotte.*)

Arnaud Delmotte, Kenichiro Tanaka, Takuya Funatomi, and Yasuhiro Mukaigawa are with the Optical Media Interface Laboratory, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan (e-mail: delmotte\_ar@yahoo.fr; ktanaka@is.naist.jp; funatomi@is.naist.jp; mukaigawa@is.naist.jp).

Hiroyuki Kubo is with the Tokai University, 2-3-23 Takanawa, Minato, Tokyo 108-8619, Japan (e-mail: hkubo@tsc.u-tokai.ac.jp).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TMM.2020.3025660>.

Digital Object Identifier 10.1109/TMM.2020.3025660

3D scanning technology has become more affordable, with free or cheap photometric reconstruction software and depth cameras. Some high-end smartphones have begun to include depth cameras that can perform 3D scans.

Due to digital rights concerns for 3D objects, 3D watermarking technology plays an important role. A user can perform a 3D scan of an object and upload the reconstructed mesh or distribute a copy of the physical object, so it would be useful to be able to prove ownership or identify the leak source. Embedding the author ID inside the model provides a proof of ownership, which is useful if someone other than the author tries to distribute it as his own.

By embedding a different ID in each object sold or provided under a non-disclosure agreement (NDA), we can identify users who illegally distribute the model. This process, called traitor tracing, is often used in other media. From a security point of view, printers could embed information such as printer ID, user ID, and printing time, similar to standard 2D paper printers [3], [4]. This could aid crime investigation when 3D printed objects, such as TSA master keys, handcuff keys, car keys, or weapons are found at a crime scene [5], [6].

In the smart manufacturing field [7]–[9], it could replace RFID tags or barcodes for identifying individual parts during the manufacturing process and embedding metadata into the objects. If 3D printer and scanner manufacturers agreed on a common watermarking method, it would also be possible to produce a “no-copy” flag that would prevent the print or scan of copyrighted objects. The printer would refuse to print if the flag is detected, similar to standard 2D printers refusing to print currency [10].

All the described scenarios except the ‘smart manufacturing’ and ‘security’ cases require the watermark to be retrievable from a standard 3D scan. Because the leaker has no interest in preserving the watermark, he would not follow any specific extraction procedure but instead do a standard 3D scan with any available equipment. If the leaker is distributing physical copies printed after scanning the object, the watermarking method also must be robust to the reprint.

Blind watermarking, which does not require any data from the original model to extract the watermark, is generally preferred for copyright contexts to avoid the risk of theft [2]. The watermark can be extracted by any user directly, whereas non-blind watermarking requires a trusted authority with a database of original models. It is thus more practical to implement a ‘no-copy’ flag on the printers or allow 3D object repositories to

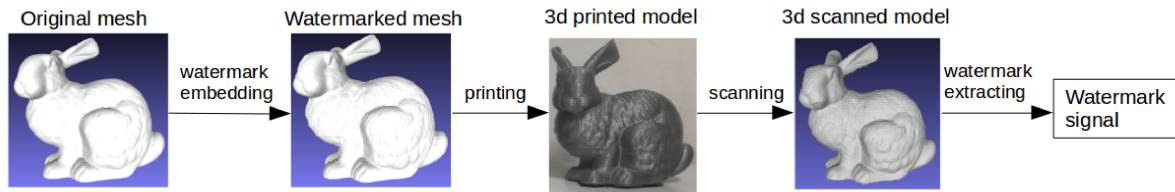


Fig. 1. 3D printing watermarking process. First, a watermark is inserted into a 3D mesh, and the mesh is then printed to produce a physical object. The watermark can be retrieved from the physical object by scanning and extracting the watermark from the scanned mesh.

verify the uploaded models automatically. In a security context, if the original is not public or available, it would be impossible to decode with non-blind watermarking.

In this paper, we focus on blind watermarking that is robust to 3D printing followed by 3D scanning, as illustrated in Fig. 1. The novelties of our method are as follows.

- Extension of the vertex norm histogram to a continuous surface, making it invariant to the sampling.
- Using the 3D moments to synchronize the watermark signal in a 3D printing context.

The strengths of our method are as follows.

- Compatibility with most 3D printers and scanners with sufficiently high resolution.
- Compatibility with a wide variety of object shapes, as demonstrated using a public dataset [11].
- Persistence of the watermark signal in duplicated objects.

This paper extends our conference paper [12] that proposed a blind watermarking method robust to the print/scan process and based on the vertex norm watermarking technique [13], [14]. We extend our previous method with the following improvements.

- 1) We subdivide the mesh angularly and radially instead of only radially to increase the number of bins without becoming too thin for the printing precision.
- 2) We combine the bins nonlinearly to avoid instability where large bins receive too much weight in the bit value computation.
- 3) We apply quantization on the min-max norms to prevent local distortion from misaligning the histogram.
- 4) We perform a robust evaluation using a dataset from a public benchmark [11], including the tolerance to alignment error and the bit error rate (BER) on simulated and actual prints.

The rest of this paper is organized as follows: In section II, we introduce related works and address our work among them. In section III, we describe the principle of our algorithm, and detail its implementation in section IV. In section V, we describe our experiments. Finally, in section VI, we discuss our results and plans for future works.

## II. RELATED WORKS

Many studies have been performed for media watermarking, such as image [15], [16], audio [17], video [18], and 3D mesh [19], [20], but it remains a relatively new topic as applied to 3D printing.

Wang *et al.* [21] and Medimegh *et al.* [22] performed comprehensive surveys of 3D mesh watermarking, and Hou *et al.*

[23] surveyed the intellectual property protection issues and solutions in the 3D printing environment. In this section, we briefly introduce the ideas behind the related works, and Table I presents a more detailed evaluation of their properties.

Most blind 3D mesh watermarking methods depend on the mesh topology and are not directly applicable to a 3D printing context because the original topology is lost during the printing and scanning processes.

For non-blind watermarking, it is sometimes possible to overcome the topology loss by aligning the scanned object to the original non-watermarked model.

Yamazaki *et al.* [24] used this approach, aligning the scanned object to the original mesh and using a spectral-domain watermarking method. Hou *et al.* [25] also developed a non-blind 3D printing watermarking method using a circular shift structure. They used the original mesh to retrieve the base axis during the extraction process. A few other mesh watermarking methods have not been tested in 3D printing contexts but are good candidates due to their resistance to resampling: Lee *et al.* [26] proposed a semi-blind mesh watermarking method using the distribution of surface normals. Theoretically, the normal distribution should be retrievable from a 3D scan after applying some filtering to remove the layering distortions from the scanned object. Lee *et al.* [27] improved their method and made it blind by including an alignment process based on rank minimization. Liu *et al.* [28] proposed a blind spectral mesh watermarking method using the Manifold Harmonics Transform (MHT). Compared to classical Laplacian matrix-based spectral analysis [29], [30], the MHT provides some resistance to resampling that may enable blind 3D printing watermarking. The main limitation is a low capacity (5 bits). To improve the embedding stability, increase the robustness and capacity, and handle more attacks such as cropping, they developed an improved method [31] using the joint geometric and texture information. However, such texture information is not available on a 3D printed object, and is therefore not applicable to our context.

For blind 3D printing watermarking, the most straightforward approach consists of printing or carving a barcode into the object. Adobe [32] patented a 3D barcode that can be added to an object during printing. Zhang *et al.* [33] used a regular grid of small bumps on a flat surface and decoded it with an RGB camera using deep learning. Harrison *et al.* [34] proposed inserting an acoustic barcode that produces an identifiable sound when swiped with a fingernail, marker, or other tools on an object's surface. HP [35] and Rize [36] proposed printing QR codes visible under UV light on an object's surface using invisible ink. Maia *et al.* proposed LayerCode [37] that prints a 1D barcode across the object using

TABLE I  
 COMPARISON OF OUR METHOD WITH RELATED WORK

	Yamazaki <i>et al.</i> [24]	Hou <i>et al.</i> [25]	Adobe [32]	Zhang <i>et al.</i> [33]	Harrison <i>et al.</i> [34]	LayerCode [37] (Dual color)	LayerCode [37] (Variable layer height)	LayerCode [37] (UV ink)	Delmotte <i>et al.</i> [38]	HP [35], Rize [36]	RFID tags [39]–[41]	Okada <i>et al.</i> [43]	Infrastructs [42]	AirCode [44]	Hou <i>et al.</i> [45]	Ours
Blind	×										✓					
Invisibility	++				--		+/-	++	+				++			+
Capacity (bits)	256	24	> 64		8-24	~ 24	~ 12	~ 64			> 64	> 25	> 64		1	16~32
Extraction from standard 3D scan			✓			Color scan	✓	×	High resolution			×				✓
Resistance to reprint	Probably		✓			Color print					×				Only if same orientation	✓
Irregular shape	✓		×			✓		Future work	?		✓		?		✓	✓
Decoding equipment	3D scanner	RGB camera	fingernail, marker, ...			RGB camera		2D document scanner	RGB camera	RGB camera	RFID tag reader	Far-infrared camera	Terahertz camera	Projector + Camera		3D scanner
Supported printers		All				Dual color printer	FDM	Dual resin printer	FDM	MJF, ADP	All, but requires insertion of an RFID tag	SLA or similar	All, but with limitations for FDM	Polyjet or similar (not FDM)	FDM	All

For ‘Invisibility,’ ‘++’ means totally invisible to the human-eye, ‘+’ means that some small artifacts are visible upon careful examination, ‘+/-’ means that some artifacts are easily visible, ‘--’ means that the watermark is clearly visible. ‘Extraction from standard 3D scan’ means that it does not require a specific procedure for the extraction besides a complete 3D scan.

dual-color printing, UV ink, or a layer thickness modification. We proposed a method [38] that embeds 2D tags by locally modifying the thickness of the printed layers.

For total imperceptibility, one solution entails inserting the watermark inside the object. Multiple patents propose to use RFID tags [39]–[41] inside a 3D object. Willis *et al.* [42], Okada *et al.* [43], and Li *et al.* [44] developed methods for embedding a tag similar to a QR code below the surface of an object and retrieving it using a terahertz camera, a thermal camera, and a projector/camera system, respectively.

Hou *et al.* [45] published a blind watermarking method robust to the print/scan process, with the most similar properties to our method. They used spread-spectrum watermarking on a set of slices. During extraction, the scanned mesh is reoriented by analyzing the layering artifacts caused by fused deposition modeling (FDM) printing to find the printing axis. However, the capacity is limited to a single bit, and it does not work without the layering artifacts, which are undetectable if the scanner has a lower resolution than the printer. It also does not allow reprinting the model with a different orientation than that of the watermark.

In a related field, Li *et al.* [6] proposed a fingerprinting method that enables signature extraction from an object to identify the printer that produced the object.

We compare our method with the related works in Table I.

### III. CORE IDEAS OF THE PROPOSED METHOD

The proposed method is an extension of our previous paper [12], which was based on the 3D mesh watermarking method from Cho *et al.* [13], [14], that used a distribution of vertex norms. Cho *et al.*’s method was developed for digital data with a mesh structure, not for the print/scan scenario. It has a significant advantage due to its invariance to translation, rotation, rescaling, and vertex connectivity. However, it can also have problems when resampling is applied, which is the case in a 3D print/scan. In this section, we first briefly explain the original method from

Cho *et al.* [13], [14] in Sec. III-A, then explain our modifications for 3D print/scan. Our first contribution (Sec. III-B) is a modification of the norm histogram calculation method to make it invariant to the sampling. This method is done by integrating the norm continuously on the surface instead of calculating the average norm of the individual vertices. Our second contribution (Sec. III-C) consists of a more robust estimation of the center of the object and its minimum and maximum norms to prevent misalignment in the watermark extraction. The center estimation is done by a 3D moment-based method that has been proven to be more robust in the 3D printing context than the classical average of the vertices [45]. The minimum and maximum norms are quantized to be more resistant to local distortion. Our third contribution (Sec. III-D) is an improved method to subdivide the mesh into bins. By performing angular and radial subdivisions, we achieve high capacity and redundancy while preventing the bin size from reaching the printer resolution limit. Finally, our fourth contribution (Sec. III-E) is a bin combination method that efficiently selects the bins used for calculating each bit value. Our method ensures the necessary redundancy to robustly encode the watermark bits and spread the signal over the entire surface to make it resistant to local degradation.

For convenience, we refer to the minimum and maximum norms as ‘min-max norms’ hereafter.

#### A. Watermarking Using Distribution of Vertex Norms [13], [14]

The mesh watermarking method that we extended in this paper is based on a histogram of vertex norms. Each vertex  $V_i$  of the mesh can be represented in spherical coordinates  $(\rho_i, \theta_i, \phi_i)$ , where  $\rho_i$  is the norm of the vertex, that is, the distance between the vertex and the center of the object, and  $(\theta_i, \phi_i)$  are the direction angles. By finding all the vertex norms  $\rho$ , a histogram of the norm distribution can be created (Fig. 2(a)). The histogram is then cropped to retain only the range between the minimum and maximum norms and subdivided into bins (Fig. 2(b)). The vertical lines represent the separation between bins. One bit is

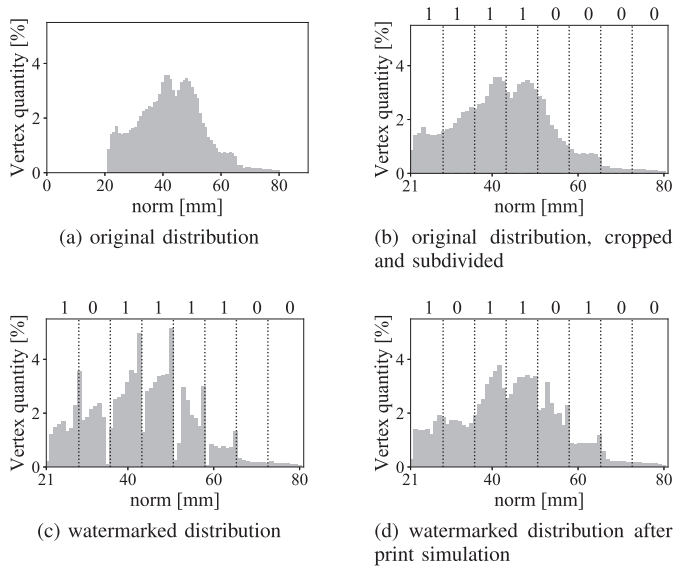


Fig. 2. Distribution of vertex norms from the Stanford bunny model, each vertical striped line represent the border of a bin. (a) Full histogram before watermarking. (b) Histogram before watermarking, cropped to keep only the range of the min-max norms, and subdivided into eight bins. (c) Histogram after watermarking, watermark value = 10111000. (d) Histogram after printing simulation.

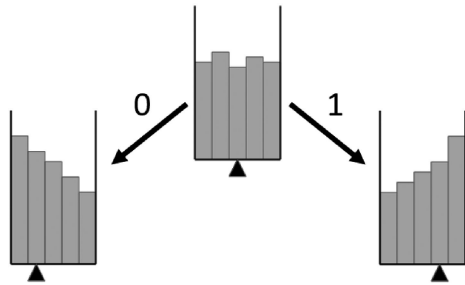


Fig. 3. Encoding one bit of data by shifting the mean of the distribution to the left or right side of the bin. The black triangle cursors below each bin represent the position of the mean of the distribution.

embedded inside each bin by modifying the norm distribution such that the mean of the distribution is on the left or the right of the center of the bin, to encode a 0 or 1, respectively, as shown in Fig. 3. To modify the norms distribution, each vertex norm  $\rho_i$  is increased or decreased, without exceeding the bounds of its bin. The modification amplitude is kept low enough such that the shape deformation is imperceptible to the human eye, and the directions  $(\theta_i, \phi_i)$  remain unchanged. Figure 2(c) shows an example of a histogram after embedding a watermark.

### B. Increasing Robustness Against Resampling

The original method uses the vertex norms to encode the watermark, which are usable for mesh watermarking but not for 3D printing. The print/scan process produces a complete resampling of the mesh and causes decoding errors because the vertex positions in which the watermark has been encoded are lost, and new vertices are generated. Using a uniform sampling with high

vertex density could mitigate this effect, but the watermarking process does not preserve the uniformity.

To solve this problem, we compute the norm histogram continuously over the entire surface instead of for a discrete set of vertices. It is a continuous rather than discrete approach and is therefore insensitive to resampling if the shape is preserved.

In practice, to compute the average norm in each bin, each triangle is first subdivided such that each sub-triangle fits completely inside one bin. Then, we compute the average norm of the surface of all the triangles contained in each bin.

$$Surf(V_A, V_B, V_C) = \frac{\|\vec{V}_A \vec{V}_B \times \vec{V}_A \vec{V}_C\|}{2} \quad (1)$$

$$A = \sum_{(V_A, V_B, V_C) \in Bin} Surf(V_A, V_B, V_C) \quad (2)$$

$$\mu = \sum_{(V_A, V_B, V_C) \in Bin} \frac{2Surf(V_A, V_B, V_C)}{A} \int_0^1 \int_0^{1-\lambda_2} \|\lambda_1 V_A + \lambda_2 V_B + (1 - \lambda_1 - \lambda_2)V_C\| d\lambda_1 d\lambda_2 \quad (3)$$

where  $V_A$ ,  $V_B$ , and  $V_C$  are the three vertices of a triangle in the bin,  $Surf$  is the function that computes the area of a triangle,  $A$  is the total area of the bin, and  $\mu$  is the average norm of the bin. The integration of the surface norms on 3D triangles (Eq. 3) is difficult to calculate analytically because of the L2-norm operator in the integral. Instead, we approximate it numerically by the average of the norm of the three vertices multiplied by the area of the triangle.

$$\mu \approx \sum_{(V_A, V_B, V_C) \in Bin} \frac{\|V_A\| + \|V_B\| + \|V_C\|}{3} \frac{Surf(V_A, V_B, V_C)}{A} \quad (4)$$

To obtain a high-precision numerical approximation, each triangle must be subdivided until the difference between the norm of the center and the mean of the norms of the three vertices becomes negligible. This condition is expressed as

$$\left| \left\| \frac{V_A + V_B + V_C}{3} \right\| - \frac{\|V_A\| + \|V_B\| + \|V_C\|}{3} \right| \leq \epsilon_c, \quad (5)$$

where  $\epsilon_c$  denotes the desired precision.  $\epsilon_c$  is expressed in the same unit as the coordinates of the vertices, which is millimeters in this case. Section V-A evaluates the precision based on the  $\epsilon_c$  value.

### C. Improving the Robustness to Misalignment

To extract the watermark without errors, we must ensure that we do not have any misalignment during the decoding process. Any error in the center position or the min-max values would affect the surface norms estimation or shift the bins. Section III-C1 and III-C2 explain how we compute more robustly the center of the object and the min-max norms, respectively.

1) *Center Position Estimation:* As explained in [46], center estimation is a critical part of the method. If the center position changes, the norms also change, and the watermark may not be extracted correctly. Because the mean of the vertices is not resistant to resampling, we use a moment-based center estimation that has been proven to be more resistant in the 3D printing

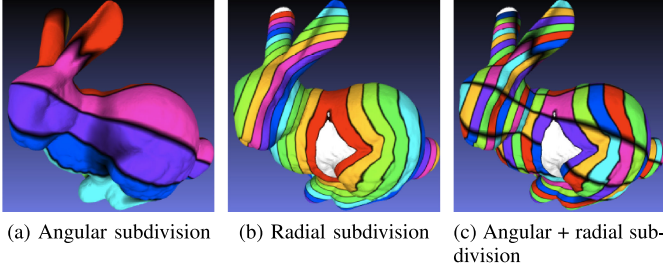


Fig. 4. Mesh subdivision into bins. (a) Angular subdivision into eight bins. (b) Radial subdivision into 16 bins + two border regions (in white). (c) Combined angular and radial subdivision into 128 bins + two border regions (in white).

context [45].

$$c = (\bar{x}, \bar{y}, \bar{z}) = \frac{(M_{100}, M_{010}, M_{001})}{M_{000}}, \quad (6)$$

$$M_{pqr} = \iiint x^p y^q z^r \tau(x, y, z) dx dy dz \quad (7)$$

$$\tau(x, y, z) = \begin{cases} 1, & \text{if } (x, y, z) \text{ is inside the mesh} \\ 0, & \text{otherwise} \end{cases}$$

where  $c$  is the estimated center,  $M_{pqr}$  denotes the  $p$ ,  $q$ ,  $r$ -th order volume moment of the mesh, and  $\tau(x, y, z)$  is an indicator function. The 3D moments can be computed efficiently using the method described in [47].

The watermarking process slightly modifies the center of gravity, the min-max norms, and the orientation. To avoid any problem due to the modifications, we recompute these parameters at each iteration of the optimization process, which is explained later, until convergence.

2) *Min-Max Norm Quantization*: On the object surface, the regions that are within a few tenths of millimeters of the min-max norms are the most sensitive to errors. Any distortion can modify the min-max norm values and result in a shift of the histogram. To prevent this problem, we quantize the min-max norms instead of using a continuous value. The shape is first slightly modified such that the min-max norms are an integer multiple of  $Q_{norm}$  during the watermarking process, with  $Q_{norm}$ , the quantization step. During extraction, we round the measured min-max norms to the closest quantized value, which provides resistance to any error strictly lower than  $Q_{norm}/2$ . To remain robust to uniform scaling,  $Q_{norm}$  is defined as a value proportional to the cube root of the object's volume.

In this paper, the quantized min-max norms are written as  $\lfloor \mathcal{N} \rfloor$  and  $\lceil \mathcal{N} \rceil$ , respectively.  $\Delta \mathcal{N}$  is equal to  $\lceil \mathcal{N} \rceil - \lfloor \mathcal{N} \rfloor$ .

#### D. Mesh Subdivision Into Bins

The bin subdivision consists of selecting the portions of the surface used for each bin. This is a delicate task because any misalignment during the extraction process could strongly affect the decoded value. As illustrated in Fig. 4, we use two subdivision methods simultaneously to cut the mesh into slices along different directions. The angular subdivision (Fig. 4(a)) is the

main extension to our previous paper. It enables increasing the number of bins without producing slices so thin that they would approach the precision limits of the 3D printer and 3D scanner.

1) *Radial Subdivision*: The radial subdivision is similar to the original vertex norm watermarking method [13], [14]: we compute the norm for each point on the surface with  $\rho = \sqrt{(x - \bar{x})^2 + (y - \bar{y})^2 + (z - \bar{z})^2}$ , where  $(\bar{x}, \bar{y}, \bar{z})$  is the center of the object (Section III-C1).

We search for the min-max norms, apply the quantization (Section III-C2), leave a margin  $\delta_{\mathcal{N}}$  around the quantized min-max norms, and then subdivide the remaining range into  $N_r$  equal slices. The size of a slice is computed by :

$$S_r = \frac{(1 - 2\delta_{\mathcal{N}})\Delta \mathcal{N}}{N_r} \quad (8)$$

We generally use  $\delta_{\mathcal{N}} = 0.05\Delta \mathcal{N}$  to prevent the min-max norms from modification during watermark embedding.

2) *Angular Subdivision*: Compared to the radial subdivision, the angular subdivision method is not invariant to rotation, and we need to align to a standard orientation. We use to align the principal axes to the base axes of the coordinate system [47]. This is done by first centering the model (Section III-C1). Then, the principal axis can be obtained by applying to the second-order moment matrix :

$$\begin{bmatrix} M_{200} & M_{110} & M_{101} \\ M_{110} & M_{020} & M_{011} \\ M_{101} & M_{011} & M_{002} \end{bmatrix} \quad (9)$$

where  $M_{pqr}$  denotes the  $p$ ,  $q$ ,  $r$ -th order moment of the mesh. To make the result unique, the third principal axis must be

the cross product of the first and second PCA axes and the third-order moments  $M_{300}$  and  $M_{030}$  must be positive [47]. The mesh is then rotated to align its principal axis to the global axes. Each vertex is transformed by :

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} & \mathcal{A}_{13} \\ \mathcal{A}_{21} & \mathcal{A}_{22} & \mathcal{A}_{23} \\ \mathcal{A}_{31} & \mathcal{A}_{32} & \mathcal{A}_{33} \end{bmatrix} \begin{bmatrix} x - \bar{x} \\ y - \bar{y} \\ z - \bar{z} \end{bmatrix} \quad (10)$$

where  $(x', y', z')$  is the centered and reoriented vertex,  $\mathcal{A}$  is the eigenvector matrix of the second-order moment matrix,  $(x, y, z)$  is the vertex before centering and reorienting, and  $(\bar{x}, \bar{y}, \bar{z})$  is the center of the object (Section III-C1).

Finally, we apply the subdivision by computing for each vertex the angle  $\Phi = \text{atan2}(z', y')$ , and subdivide the mesh into bins of size  $S_a = \frac{2\pi}{N_a}$  rad on the angle  $\Phi$ , where  $N_a$  denotes the number of angular subdivisions.

This alignment method does not work if the mesh has rotational symmetry around an axis. For this type of mesh, we need to restrict  $N_a$  to 1 in our algorithm, which means we use only the radial subdivision.

#### E. Combination of Multiple Non-Consecutive Bins

To achieve a good resistance against local printing artifacts and scanning errors, we combine the value of multiple bins to encode each watermark bit. The signal is spread over the entire surface to prevent sensitivity to local error. The spreading is done

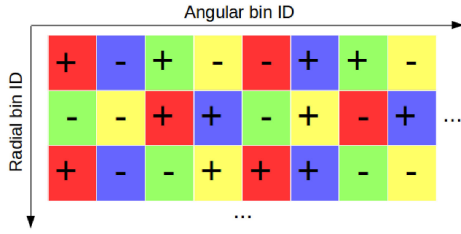


Fig. 5. Bin lookup table example with four bits: each cell corresponds to one bin ID, each color corresponds to one bit of the watermark sequence, and each “+” or “-” corresponds to the bin’s invert flag.

via a fixed-function independent of the mesh shape to prevent problems during extraction. To determine which bins contribute to encode a watermark bit, we use a  $N_a \times N_r$  lookup table that contains a bit index and an invert flag for each bin. For each bin  $(i, j)$ , the bit index  $binId_{i,j}$  defines which watermark bit is encoded, and the invert flag  $binInvert_{i,j}$  is a Boolean that, if True, inverts the encoded value by applying a NOT operator. The bit indices are chosen such that neighboring bins do not encode the same bit, and the invert flags are chosen by a pseudo-random sequence with a known seed retrievable during extraction.

The invert flags reduce the correlation between bins with the same  $binId$ . They reduce the sensitivity to angular misalignment because the misaligned portions become uncorrelated, and also aid embedding by decreasing the initial strength of the bins (Section V-C).

The watermark bit values are computed by a weighted average of the bins. More weight is given to the large surface bins, without making the small bins negligible. Our weight function consists of an identity function of the surface area until it reaches the mean area  $\bar{A}$ , at which point we multiply the excess by 0.25:

$$weight(a, \bar{A}) = \begin{cases} a, & \text{if } a < \bar{A} \\ \frac{3\bar{A}+a}{4}, & \text{otherwise} \end{cases} \quad (11)$$

where  $a$  is the surface area of the bin, and  $\bar{A}$  is the surface area of the entire object divided by the number of bins ( $N_a N_r$ ).

Algorithm 1 shows how  $binId$  and  $binInvert$  are used to combine the bins into the watermark signal, Algorithm 2 describes how the lookup table is generated, and Fig. 5 gives an example of a generated table.

#### IV. WATERMARK EMBEDDING AND EXTRACTION ALGORITHM IMPLEMENTATION

In this section, we describe the algorithm used to embed and extract the watermark. Section IV-A describes the extraction algorithm and Section IV-B describes the embedding algorithm.

The extraction algorithm is used inside the embedding algorithm, as illustrated in Fig. 7, so we describe it first.

##### A. Watermark Extraction Algorithm

As illustrated in Fig. 6, the extraction algorithm consists of computing the histogram of surface norms for each bin, and then combining the bins to compute the mean for each bit. Each bit is set to 1 if its histogram mean is greater than 0.5, or 0 otherwise.

Specifically, we begin by centering the model with the moment method described in Section III-C1, and performing the rotation alignment described in Section III-D2. Next, we subdivide each triangle such that the three vertices are fully included in a single bin and until it matches the criterion from Eq. 5 in Section III-B.

We then define a function that remaps a norm  $X$  to the range  $[0,1]$ , with 0 and 1 corresponding to the minimum and maximum radial boundary of the bin that contains it, respectively :

$$f_j(X) = \frac{X - (\lfloor \mathcal{N} \rfloor + \delta_{\mathcal{N}} \Delta \mathcal{N} + j S_r)}{S_r} \quad (12)$$

where  $\lfloor \mathcal{N} \rfloor$  is the quantized minimum norm (Section III-C2),  $\delta_{\mathcal{N}}$  is the margin around the min-max norm (Section III-D1),  $\Delta \mathcal{N}$  is the difference between the minimum and maximum norms (Section III-C2),  $j$  is the index of the radial slice, and  $S_r$  is the size of the radial slice (Section III-D1). By using Equations 2 and 4 from Section III-B combined with the remapping function (Eq. 12), we compute the total area  $A_{i,j}$  and mean norm  $\mu_{i,j}$  for each bin  $(i, j)$  :

$$\begin{aligned} A_{i,j} &= \sum_{(V_A, V_B, V_C) \in T_{i,j}} Surf(V_A, V_B, V_C), \\ \mu_{i,j} &= \sum_{(V_A, V_B, V_C) \in T_{i,j}} f_j \left( \frac{\|V_A\| + \|V_B\| + \|V_C\|}{3} \right) \\ &\quad \times \frac{Surf(V_A, V_B, V_C)}{A_{i,j}}, \end{aligned} \quad (13)$$

where  $V_A$ ,  $V_B$ , and  $V_C$  are the coordinates of a vertex and  $T_{i,j}$  is the list of triangles included in the bin  $(i, j)$ .

The bin values are then combined to produce the vector  $\mu'$  of size  $N_b$ , the number of bits of the watermark signal :

$$\mu' = binMerge(\mu, A) \quad (14)$$

with  $binMerge$  corresponding to Algorithm 1, explained in Section III-E. The parameters  $binId$  and  $binInvert$  used in Algorithm 1 are generated by Algorithm 2.

Finally, we extract the watermark bits by thresholding :

$$w'_k = (\mu'_k > 0.5) \quad (15)$$

##### B. Watermark Embedding Algorithm

The embedding algorithm is illustrated in Fig. 7 and described in Algorithm 3. It is an iterative process that applies gradient descent on each vertex until all the watermark bits have been encoded with sufficient strength.

Before starting the optimization, the mesh is first resampled to achieve a relatively uniform and well-adapted vertex density. If the mesh has triangles that are too large, especially if they are larger than the bin size, it is impossible to precisely modify the surface norm histogram, making it difficult to embed a watermark. Conversely, if the meshing is too detailed, the optimization process can produce a rough surface and a lower watermark strength. In practice, we apply mesh simplification with tools such as MeshLab [48] if the mesh resolution is too high, or triangle subdivision if the resolution is too low. For each edge of the mesh, we split it into two equal parts if the edge length is

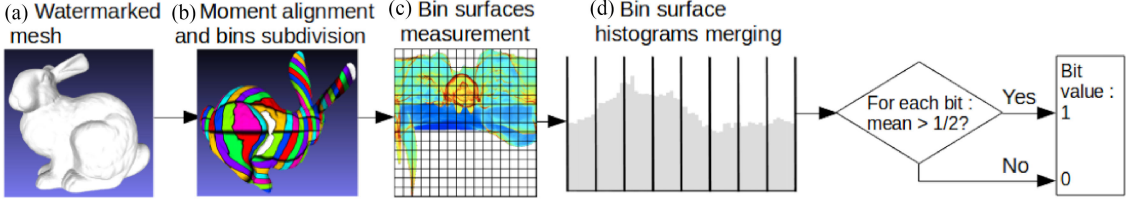


Fig. 6. Overview of the watermark extraction algorithm. (a) The mesh from which we want to extract the watermark. (b) The mesh after 3D moment-based alignment, and bin subdivision. Bins with the same color are assigned to the same watermark bit. (c) The surface distribution is computed inside each bin, with white for no surface, and from blue to red for low to high quantity. (d) The histogram formed by the combination of the bins. Each column corresponds to one bit, which has a value of 1 if the distribution is more concentrated on the right side of the bin, or 0 otherwise.

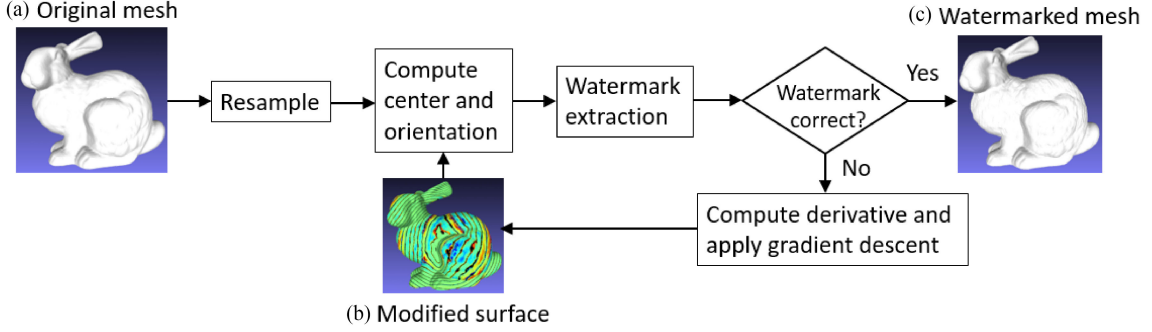


Fig. 7. Overview of the watermark embedding algorithm. (a) The mesh before watermarking (b) The shape modifications, with colors from red to blue when the surface is moved in the direction of the normal or on the opposite direction, respectively. Green indicates little to no modification. (c) The resulting mesh after the watermarking process.

---

**Algorithm 1: Bin Merging Algorithm**


---

**input :** The number of angular bins  $N_a$ , radial bins  $N_r$  and bits  $N_b$ . The mean table  $\mu$ , area table  $A$ , bit index table  $binId_{i,j}$  and bin invert table  $binInvert_{i,j}$  of size  $N_a \times N_r$

**output:** The mean array  $\mu'$  and area array  $A'$  of size  $N_b$

```

 $\mu' := [0, \dots, 0];$ 
 $U' := [0, \dots, 0];$ 
 $A' := [0, \dots, 0];$ 
 $\bar{A} := average(A);$ 
for  $i$  in  $[0, N_a]$  do
    for  $j$  in  $[0, N_r]$  do
         $k = binId_{i,j};$ 
         $W = weight(A_{i,j}, \bar{A});$  // Eq. 11
        if  $binInvert_{i,j}$  then
             $U'_k += (1 - \mu_{i,j}) * W$ 
        else
             $U'_k += \mu_{i,j} * W$ 
        end
         $A'_k += W;$ 
    end
end
 $\mu' = U' / A'$ 
    
```

---

larger than the threshold  $\epsilon_e$  or if the difference of the norm of the two vertices is larger than the threshold  $\epsilon_n$ . For stable encoding,  $\epsilon_n$  should be smaller than  $\Delta\mathcal{N}/N_r$  to ensure that the triangles

---

**Algorithm 2: Bin-Bit Assignment Algorithm**


---

**input :** The number of angular bins  $N_a$ , radial bins  $N_r$  and bits  $N_b$ .

**output:** The bit index table  $binId_{i,j}$  and bin invert table  $binInvert_{i,j}$  of size  $N_a \times N_r$

```

srand(1); // Seed for the random function
Set  $currentBit := 0;$ 
for  $j$  in  $[0, N_r]$  do
    for  $i$  in  $[0, N_a]$  do
         $binId_{i,j} := currentBit;$ 
         $binInvert_{i,j} := (rand()\%2 = 1);$ 
         $currentBit := (currentBit + 1)\%N_b;$ 
    end
    if  $((j + 1) * N_a)\%N_b = 0$  then
         $currentBit := (currentBit + N_b/4 + 1)\%N_b$ 
    end
end
    
```

---

are smaller than the radial size of the bins. However, too low a value of  $\epsilon_n$  tends to produce nearly degenerated triangles, with one angle close to  $180^\circ$ , because the triangles are cut only along the radial direction. Therefore, we should also choose a value of  $\epsilon_e$  close to the value of  $\epsilon_n$  to regularize the shape of the triangles.

Then we start the iterative process of the gradient descent. At the start of each iteration, we recompute the center of gravity, orientation, and min-max norms (Section III-C1, III-C2, III-D2).

**Algorithm 3:** Watermark Embedding Algorithm

**input :** The list of vertices  $V$  and triangles of the mesh to watermark, the  $N_b$  bit watermark sequence  $w$ , the optimization rate  $\lambda$ , and all the other watermark parameters used in the watermark extraction and embedding algorithms

**output:** The watermarked mesh obtained by the optimization process

Resample the mesh (Sec. IV-B);

Set  $N_V$ , the number of vertices  $V$ ;

Compute  $\vec{N}_v$ , the normal for each vertex  $v$ , obtained by averaging the normal of the triangles incident to the vertex  $v$ ;

**foreach**  $iter$  in  $range(0, nbIter)$  **do**

Center, reorient the mesh (Sec. III-C1 and III-D2) and apply the same rotation to each normal  $\vec{N}_v$ ;

Compute  $\lfloor \mathcal{N} \rfloor$  and  $\lceil \mathcal{N} \rceil$  (Sec. III-D1);

Compute  $\mu_{i,j}$  and  $A_{i,j}$  for each bin ; // Eq. 13

Set  $\mu' = binMerge(\mu, A)$  ; // Eq. 14

Set  $\Delta\mu := []$ ;

**foreach**  $k$  in  $range(0, N_b)$  **do**

**if**  $w_k = 1$  **then**

$\Delta\mu_k := \max(0, \mathcal{S}_{enc} - 2(\mu'_k - 0.5))$

**else**

$\Delta\mu_k := -\max(0, \mathcal{S}_{enc} - 2(0.5 - \mu'_k))$

**end**

**end**

Set  $\Delta V$ , a  $N_V \times N_b$  matrix, in which  $\Delta V_{v,k}$  represents the gradient of  $\mu'_k$  when moving the vertex  $v$  along its normal  $\vec{N}_v$ ;

**foreach**  $v$  in  $V$  **do**

$V_v := V_v - \lambda(\Delta V_{v,:} \cdot \Delta\mu)\vec{N}_v$ ;

**end**

**end**

The cost function for the gradient descent is defined by:

$$F = \sum_{k=0}^{N_b} \max(0, \mathcal{S}_{enc} - \mathcal{S}_k)^2 \quad (16)$$

where  $F$  is the cost function,  $\mathcal{S}_{enc}$  is the encoding strength, and  $\mathcal{S}_k$  is the current encoded strength of the  $k^{th}$  bit defined as follows:

$$\mathcal{S}_k = \begin{cases} 2(\mu'_k - 0.5), & \text{if } w_k = 1 \\ 2(0.5 - \mu'_k), & \text{if } w_k = 0 \end{cases} \quad (17)$$

Other than a training process or performance evaluation, the ground-truth value  $w_k$  is not available. In that case, we approximate the strength by replacing  $w_k$  by  $w'_k$  in Eq. 17, which results in an absolute value of the strength.

Finally, the minimum strength  $\lfloor \mathcal{S} \rfloor$  and the mean strength  $\bar{\mathcal{S}}$  of the watermark signal, used for performance evaluation, are defined by :

$$\lfloor \mathcal{S} \rfloor = \min_k \mathcal{S}_k \quad (18)$$



Fig. 8. The experimental dataset [11].

$$\bar{\mathcal{S}} = \frac{1}{N_b} \sum_{k=0}^{N_b} \mathcal{S}_k \quad (19)$$

The strength metrics are used during the training process and for performance evaluation but can also be used in real situations to estimate the confidence of the result. High values of both  $\lfloor \mathcal{S} \rfloor$  and  $\bar{\mathcal{S}}$  provide good confidence in the correctness of the result.

To choose the best encoding strength  $\mathcal{S}_{enc}$  and sampling parameters  $\epsilon_n$  and  $\epsilon_e$ , we evaluate multiple sets of values for these parameters, encode the watermark for each set and keep the one that has the highest value for the minimum strength  $\lfloor \mathcal{S} \rfloor$ . We can use the maximum root mean square error (MRMS) and mesh structural distortion measure (MSDM) metrics to define additional deformation and visibility thresholds [49]. The encoding strength and sampling parameters are not required for decoding, and can thus be freely chosen during the encoding process. Typical values are  $\mathcal{S}_{enc} = 5\%$ ,  $\epsilon_e = \Delta\mathcal{N}/\{32, 64\}$  and  $\epsilon_n = \Delta\mathcal{N}/\{64, 128\}$ .

## V. EXPERIMENTS

To evaluate the performances of our method, we tested the influence of our parameters and the resistance to multiple attacks. We evaluated some parameters specific to our algorithm, such as a misalignment of the center of gravity, the rotation alignment, and the min-max norms. Because our target is 3D printing, we evaluated with a 3D print simulation and with several real printed objects. Except when specified differently, we used the ten meshes from the dataset proposed in [11], illustrated in Fig. 8, with five different watermark values for each, and reported the average result.



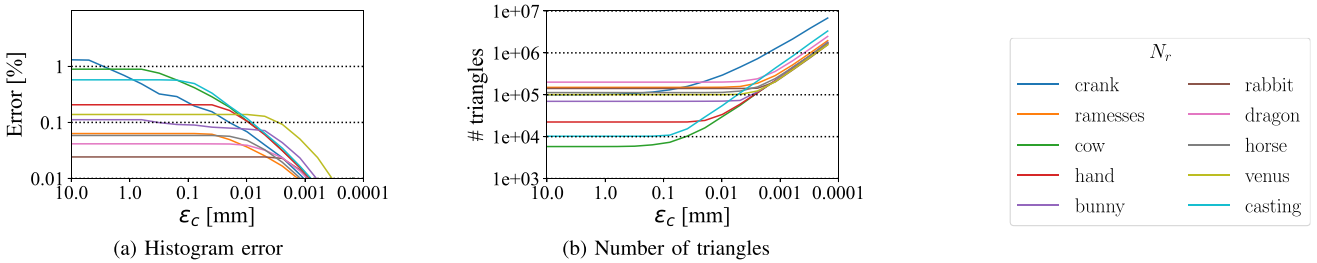


Fig. 9. Influence of the subdivision parameter  $\epsilon_c$  on the triangles' numerical integration error and on the total number of triangles. (a) Average absolute error of the bin values  $\mu_{i,j}$ , using our default parameters and multiple  $\epsilon_c$  values (lower is better). (b) Number of triangles to evaluate to compute the bin values, depending on the threshold  $\epsilon_c$  (lower is better).

TABLE II  
BOUNDING SPHERE RADIUS OF THE DATABASE MESHES AFTER UNIFORM  
SCALING TO A VOLUME OF  $86 \text{ cm}^3$

Model	Radius
Ramesses	82.0 mm
Cow	73.2 mm
Hand	72.6 mm
Bunny	61.0 mm
Rabbit	67.9 mm
Dragon	65.3 mm
Horse	84.8 mm
Venus	38.2 mm
Casting	78.0 mm
Crank	58.8 mm

We used two metrics for performance evaluation :

- Success rate, which is an all-or-nothing metric. Each evaluation provides a score of 100% if all the bits are correct, or 0% if there is at least one bit error.
- Bit-error rate (BER), which corresponds to the number of wrongly-decoded bits divided by the total number of encoded bits.

When not specified differently, we evaluated our method on the database using  $N_b = \{8, 16, 32\}$ ,  $N_a = 16$ ,  $N_r = 48$ ,  $\delta_N = 5\%$ ,  $Q_{norm} = 0.2/\sqrt[3]{86}$ ,  $\epsilon_e = \Delta\mathcal{N}/\{32, 64\}$ ,  $\epsilon_n = \Delta\mathcal{N}/\{64, 128\}$ ,  $\epsilon_c = 0.01 \text{ mm}$ ,  $\mathcal{S}_{enc} = 5\%$ . The quantization step  $Q_{norm}$  is proportional to the cube root of the volume of the object. For our default printing volume of  $86 \text{ cm}^3$ , it corresponds to a 2 mm step. Table II gives an overview of the size of the database meshes after being uniformly scaled to the default volume of  $86 \text{ cm}^3$ .

#### A. Numerical Integration of the Surface Norm

As explained in Section III-B, the integration on the triangles is done numerically instead of analytically, and the precision depends on the parameter  $\epsilon_c$ . Too high an  $\epsilon_c$  would result in errors in the histogram calculation, but too low an  $\epsilon_c$  would make it computationally expensive. This subdivision can be done triangle-by-triangle during the integration and does not require storing the full mesh at higher resolution, so it does not present a memory consumption problem. The two graphs of Fig. 9 show the effects of the subdivision with various  $\epsilon_c$  values. Figure 9(a) shows the mean histogram error, which is expressed in the same

units as the encoding strength  $\mathcal{S}_{enc}$ . Figure 9(b) shows the resulting number of triangles to process. We observe that  $\epsilon_c$  around 0.01 mm produces a negligible error if compared to an encoding strength of about 5%, and also keeps the number of subdivisions reasonable.

#### B. Resistance to Misalignment

Unlike local errors made by small printing artifacts that only affect their corresponding bins, misalignment affects all the bins. Three types of misalignment can occur in our method: on the center position, on the min-max norms, and on the orientation. In this section, we encode a watermark in the database models and then decode it with increasing shift values applied to the center position, the min-max norms, or the orientation angle. In the following subsections, we refer to the tolerance to misalignment as the highest shift value we can apply to a watermarked model without generating errors in the decoded value.

1) *Center Misalignment*: An error in the center position results in a measurement error for all the surface norms. This is the most sensitive parameter in our method. Errors in the center position are mainly caused by non-uniform scaling that may happen in a poorly-calibrated printer, by warping effects due to poor temperature control during printing and cooling, or by cropping if a part of the object breaks. It can also happen during the scanning process if the captor is poorly calibrated or if there is an error during the fusion of multiple scans. We evaluated the center misalignment tolerance using 20 random translation axes upon which we applied a shift. The result was considered incorrect if any of the 20 shifts produced an error. The results are shown in Fig. 10. We observe that the center misalignment tolerance decreases when  $N_r$  increases, because the radial slices become smaller, and the influence of the misalignment is inversely proportional to the radial size of the bins. Also, increasing  $N_a$  while keeping the other parameters constant improves the misalignment tolerance, probably because it increases the redundancy.

2) *Min-Max Norm Error*: Radial subdivision requires finding the min-max norms, and subdividing the range into a certain number of bins. An error in the min-max norm values would shift all the bin positions and produce errors in the decoded bin values. This can happen if printing or scanning artifacts occur around the min-max norm regions. We evaluated both with the

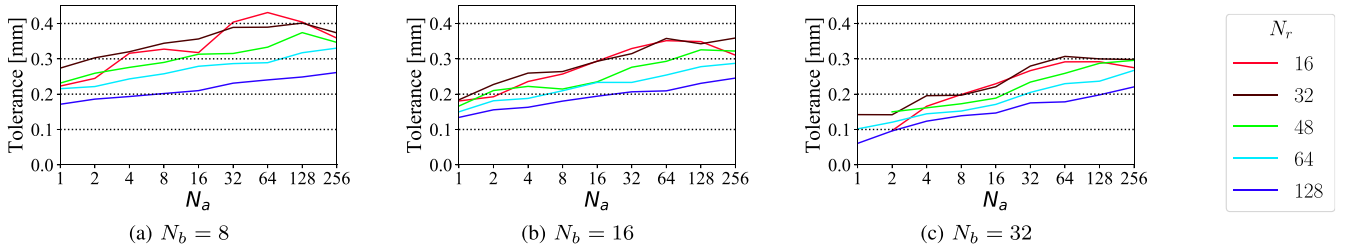


Fig. 10. Mean center position error tolerance (higher is better). The vertical axis is the average center misalignment distance that can be applied without causing any decoding error.

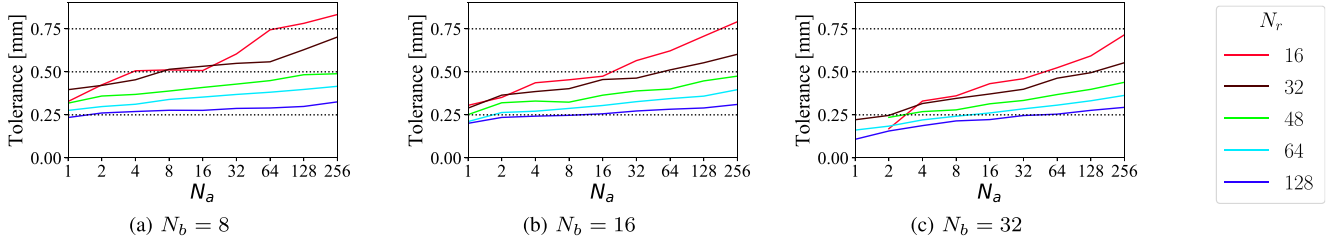


Fig. 11. Average of the min-max norm error tolerance without quantization (higher is better). The vertical axis is the average min-max norm error that can be applied without causing any decoding error, when the quantization is not used.

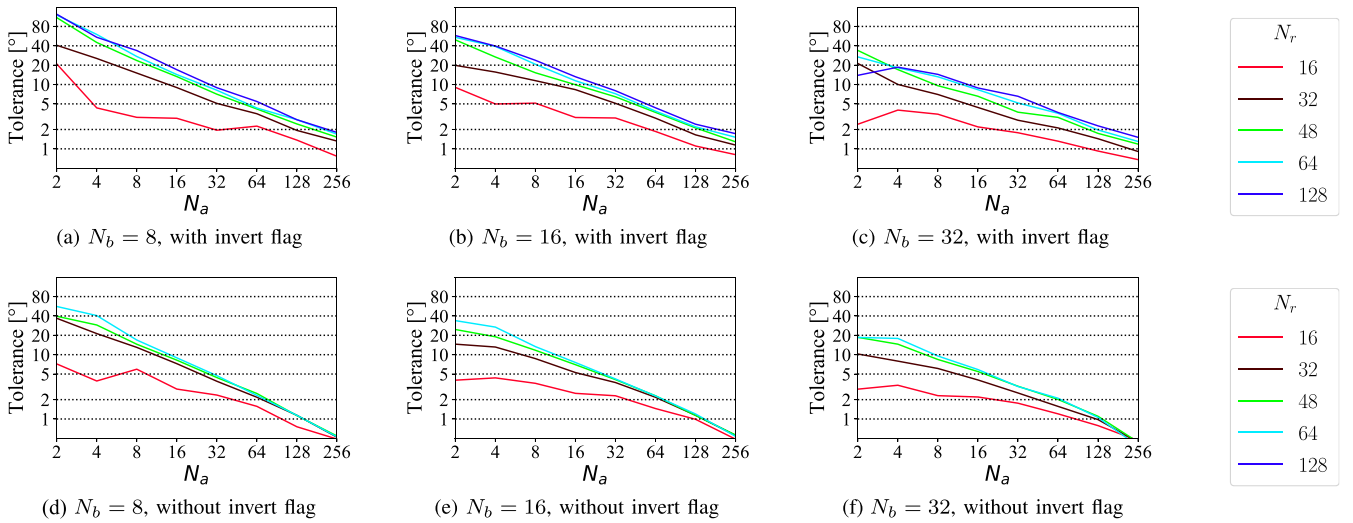


Fig. 12. Mean rotation alignment error tolerance (higher is better). The vertical axis is the average rotation misalignment distance that can be applied without causing any decoding error. The horizontal axis starts at  $N_a = 2$  because the method is invariant to rotation when  $N_a = 1$ , which would result in an infinite value.

min-max norm quantization (Section III-C2) and without. Figure 11 shows the average min-max norm error tolerance without quantization. The graphs are similar to the center misalignment tolerance (Section V-B1), and the same conclusions apply.

When we applied min-max norm quantization with our default parameters  $Q_{norm} = 0.2/\sqrt[3]{86}$  and an object volume of  $86 \text{ cm}^3$ , which results in a step of 2 mm, we obtained a constant tolerance just below 1 mm.

3) *Rotation Misalignment*: The introduction of angular subdivision in the method removed the invariance to rotation and brought the risk of errors in case of misalignment. Theoretically, the error introduced in the bin value will be proportional to the angular alignment error divided by  $S_a$ , meaning that the

smaller the angular slices are, the more sensitive we become to error by rotation misalignment. To evaluate the angular tolerance to error, we selected eight rotation axes, including three base axes and five random axes, and then chose increasing misalignment angles and decoded the watermark for each angle and rotation axes. For each mesh and watermark value, we retained the largest angle which produced no decoding error on any of the eight rotation axes. Figure 12 shows the average angle misalignment tolerance. Increasing the number of angular bins makes the method more sensitive to angle error, but increasing the number of radial bins reduces the sensitivity to angle error, below 128 bins. Also, we obtain a higher tolerance with the invert flag than without it. This can be explained by the increased

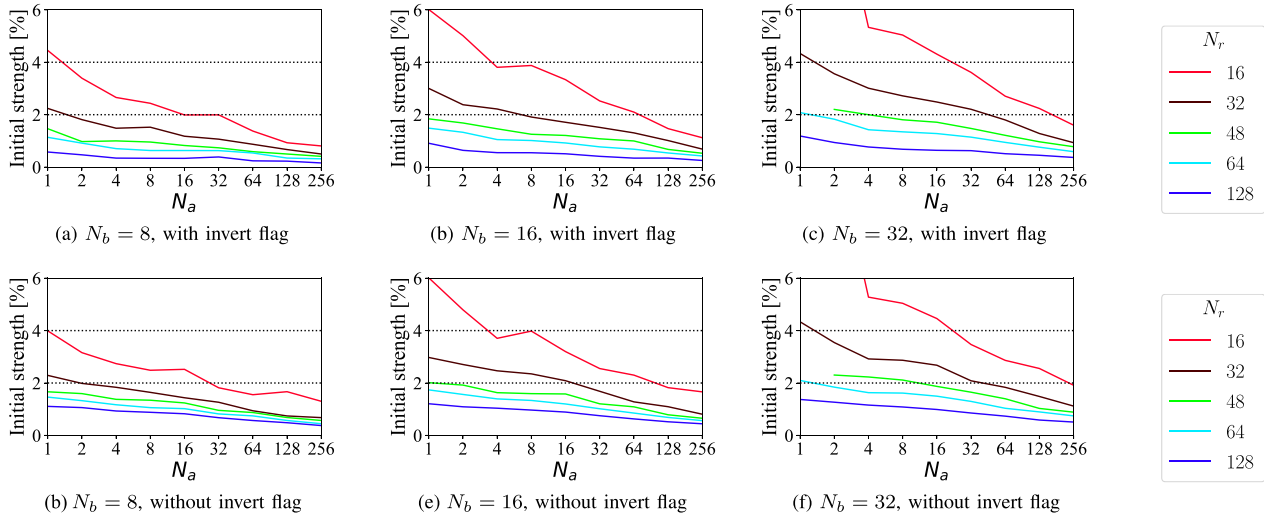
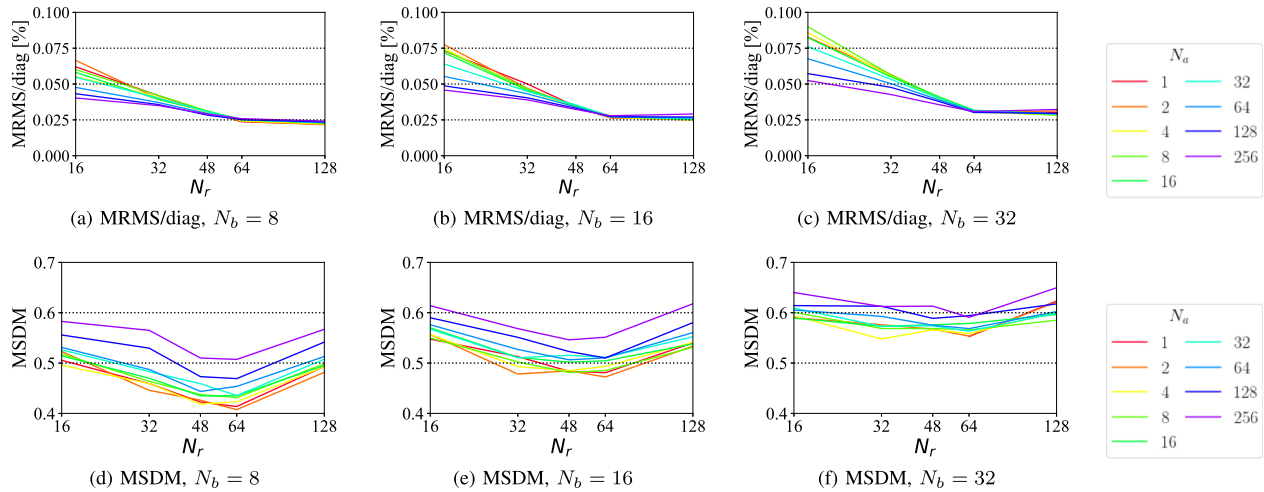


Fig. 13. Average initial strength of the bins (lower is better).


 Fig. 14. Surface degradation evaluation using the MRMS/diag and the MSDM metrics (lower is better), with  $\mathcal{S}_{enc} = 5\%$ .

redundancy and the reduced correlation of wrongly-aligned bins (Section III-E). For the configuration  $N_r = 128$  without the invert flag, the embedding process did not succeed to embed all the watermark bits in most of the evaluations. Therefore, we were not able to produce the corresponding average error tolerance curve. This happened because, for this  $N_r$  value, the triangles were as large as the bins radial size under our default sampling parameter  $\epsilon_n = \Delta\mathcal{N}/\{64, 128\}$ , which is the limit at which the encoding becomes unstable. However, the embedding was much more stable when the invert flag was used, and we successfully produced the average curves. The invert flag aids the embedding in the limit case because of the reduced correlation between neighboring bins.

### C. Initial Bin Value

Before applying the watermarking optimization process, each bin has an initial value. When this initial value is opposite to the one we need to encode, the watermarking optimization process must compensate for this value plus the additional encoding

strength. A low initial strength makes the optimization process easier and reduces the surface distortion produced by the watermarking process. In this experiment, we evaluated the influence of the invert flag and the number of radial and angular bins on the average initial strength of the bins (Fig. 13). The average initial strength decreases when the number of radial or angular bins increases and is also slightly lower with the invert flag than without.

### D. Visibility Evaluation

One of the goals of watermarking is to embed data in the model without it being visible to the human eye. We evaluated it with the same metrics as the 3D digital watermarking benchmark [11]: a geometrical metric called the maximum root mean square (MRMS) and a perceptual metric called the mesh structural distortion measure (MSDM) [49]. Figure 14 shows the influence of the parameters on the MRMS and MSDM metrics. The MRMS decreases when the number of radial bins increases, up to 64 bins, and then remains approximately constant. If the

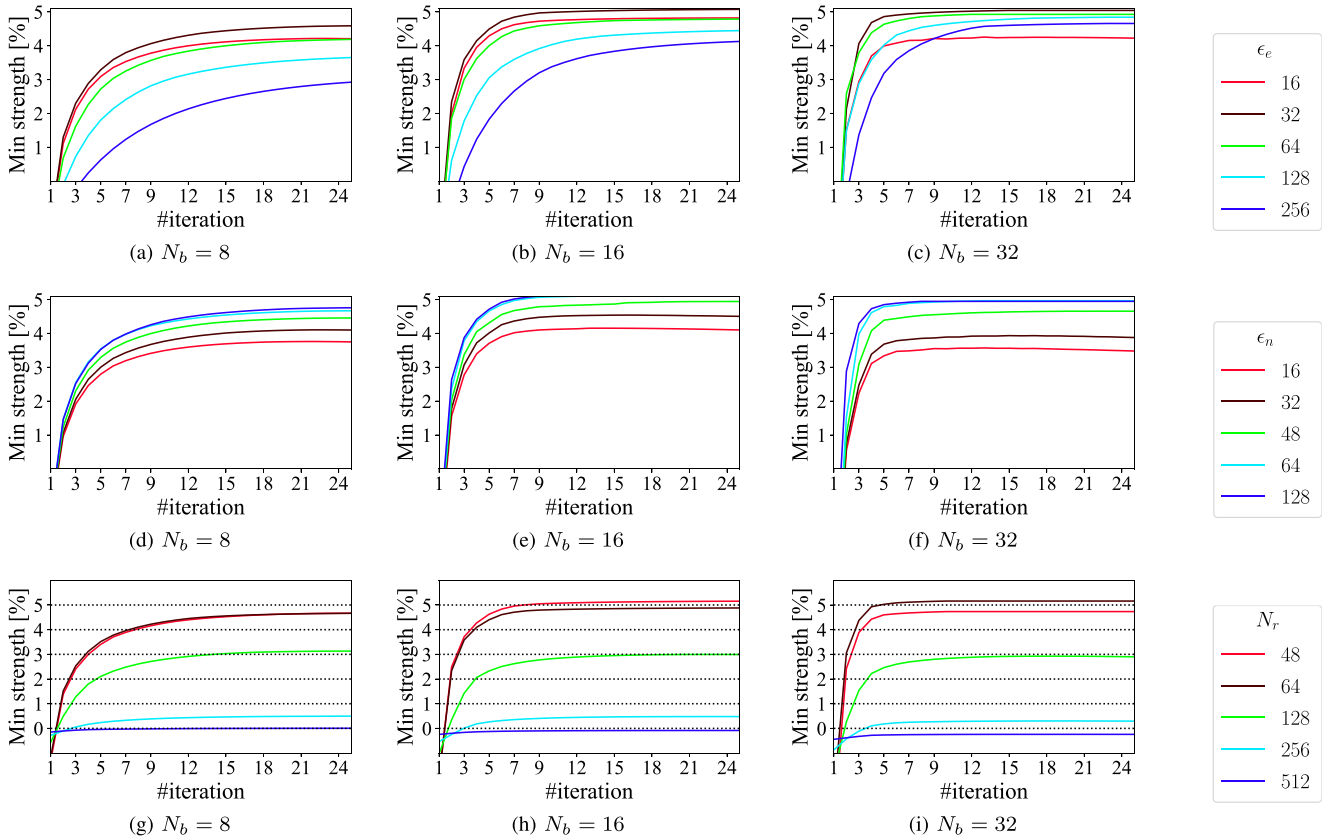


Fig. 15. Evolution of the minimum strength  $[S]$  during the embedding process (higher is better). The values are the average for all the models in the dataset, using the default parameters. The figures (a) to (c) evaluate the sampling parameter  $\epsilon_e$ , (d) to (f) evaluate the sampling parameter  $\epsilon_n$ , and (g) to (i) evaluate the number of radial bins  $N_r$ .

number of radial bins is low ( $N_r < 48$ ), increasing the number of angular bins also decreases the MRMS. The MSDM increases when the number of angular bins increases, especially for values above 32 bins. The MSDM produces a U-shaped curve for the number of radial bins, with an optimal value of approximately 48-64 bins. The difference between the two metrics can be explained by the fact that the MRMS is only influenced by the amplitude of the distortions, whereas the MSDM is also influenced by the frequency, because high-frequency distortions are more visible to the human eye compared to low-frequency distortions.

### E. Embedding Robustness Evaluation

As explained in Section IV, the sampling of the mesh and the number of radial bins affect the watermark embedding process. To analyze the robustness, we recorded the minimum embedded strength  $[S]$  for each mesh of the dataset using multiple sampling values and number of radial bins; the remaining parameters were set to default values. We plotted the average for each iteration in Fig. 15. For the sampling evaluation, we were able to correctly embed the watermark in all our experiments, but the achievable encoded strength was affected. For the threshold  $\epsilon_e$ , we obtained the best result at  $\epsilon_e = 32$ . Lower or higher values resulted in a lower minimum strength  $[S]$  on average. For

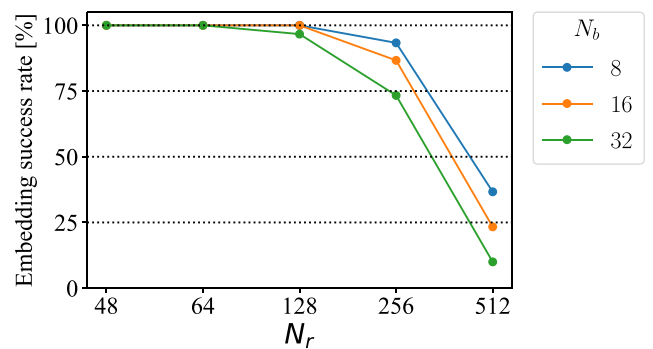


Fig. 16. Embedding success rate for multiple radial bins  $N_r$  for all the dataset models (higher is better).

the threshold  $\epsilon_n$ , we obtained the best result for either  $\epsilon_n = 64$  or  $\epsilon_n = 128$ . We did not evaluate values of  $\epsilon_n$  above 128 and values of  $\epsilon_e$  above 256 due to memory saturation. For  $N_r$  radial bins, we observed a significant decrease in the embedded strength starting from  $N_r = 128$ , even though we only had some embedding failures for 32 bits, as shown in Fig. 16. An embedding is considered failed if at least one bit contains the wrong value. Higher values of  $N_r$  resulted in more embedding failures, including for the 8- and 16-bit watermarks.

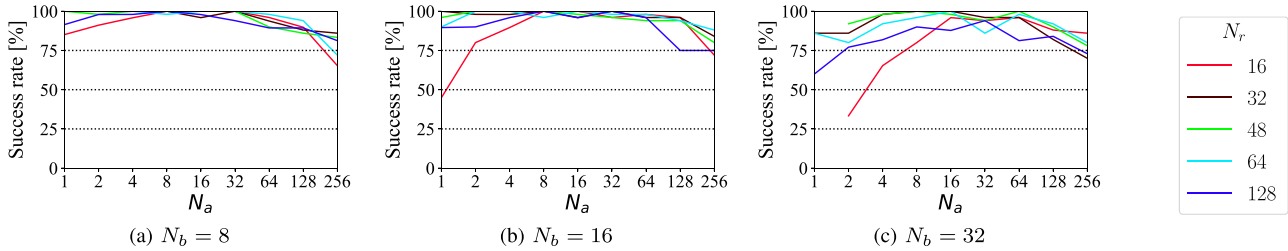


Fig. 17. Average success rate of the watermark decoding process after printing simulation with 0.3 mm layer height.

### F. 3D Print Simulation

Printing and scanning 3D objects is a time-consuming process, limiting the number of experiments that can be performed. Therefore, we developed a printing simulation process that takes the ‘gcode’ from the printer slicing software, simulates the print using the flow math proposed in [50], [51], generates a 2D truncated signed distance function (TSDF) for each layer, fills the holes inside the object, and generates the corresponding mesh with the layering artifacts using the Marching Cube algorithm. This process simulates the quantization of the shape, the rounded borders of the layers, and the resampling of the mesh, which are the main artifacts produced by a real print. We used a 0.3 mm layer height, which corresponds to the coarsest resolution generally used by consumer 3D printers. The result in Fig. 17 shows the influence of the number of radial and angular bins on the success rate. As expected, the performance decreases when the watermark length  $N_b$  increases because it reduces the redundancy. Our best range seems to be around [32,64] for  $N_r$ , and around [8,64] for  $N_a$ .

### G. Real Print-Scan

To evaluate our method in real conditions, we performed a few real print/scan experiments using the dataset objects with 8- to 32-bit watermarks. We removed the ‘crank’ model because our 3D scanner was not able to produce a complete scan due to the number of holes and occluded surfaces in the model. We used an ‘original Prusa I3 MK3S’ with white PLA to print and an ‘HP 3D Structured Light Scanner Pro S3’ to scan. Because the print/scan is very time consuming, we only did one evaluation per model and watermark size, for a total of 27 objects. Based on the simulation results for misalignment resistance, visibility, and print simulation, we chose the parameters  $N_r = 48$  and  $N_a = 16$  for the real prints. A subset of our printed objects is illustrated in Fig. 18, and the bit error rates are reported in Fig 19. To measure the surface degradation caused by the print-scan process, we aligned the scanned meshes using iterative closest point (ICP) and computed the MRMS metric divided by the diagonal of the object. All our tested models except ‘casting’ succeeded up to 16 bits, and the error rate remained low at 32 bits. We also observe that a majority of our experiments produced a surface degradation (MRMS/diag) between 0.06% and 0.2%, which is higher than the 0.025 – 0.05% degradation caused by the watermarking process (Section V-D, Fig. 14). The ‘casting’ model produced the highest error rate in our evaluation. After performing a few more experiments with this model, we discovered that



Fig. 18. Database models printed with an 8-bit watermark.

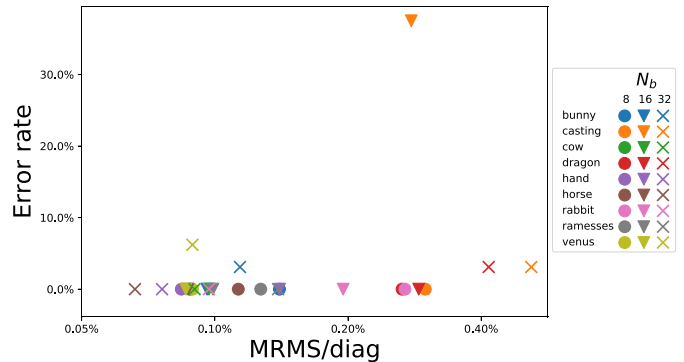


Fig. 19. Decoding error rate for real print experiments. The horizontal axis represents the degradation caused by the print/scan process, measured by the MRMS divided by the diagonal of the object.

the scanning process generally produces high distortion because of the lack of features to align and fuse multiple shots. With extreme care in scanning, we succeeded to decode it without error, but the process was much more difficult than for the other models. A higher-quality scanner would likely have fewer problems.

We performed a few more experiments to verify that our method also works with ABS instead of PLA and that it was robust to a reprint, *i.e.*, printing the model obtained after scanning a watermarked object, producing twice as many printing and scanning errors. We printed ‘bunny’ and ‘Venus’ with a 16-bit watermark using ABS without decoding errors. We reprinted ‘bunny’ and ‘Venus’ with 8- and 16-bit watermarks. We encountered a 1-bit error on ‘bunny’ with 16 bits, but had no errors on the other three prints.

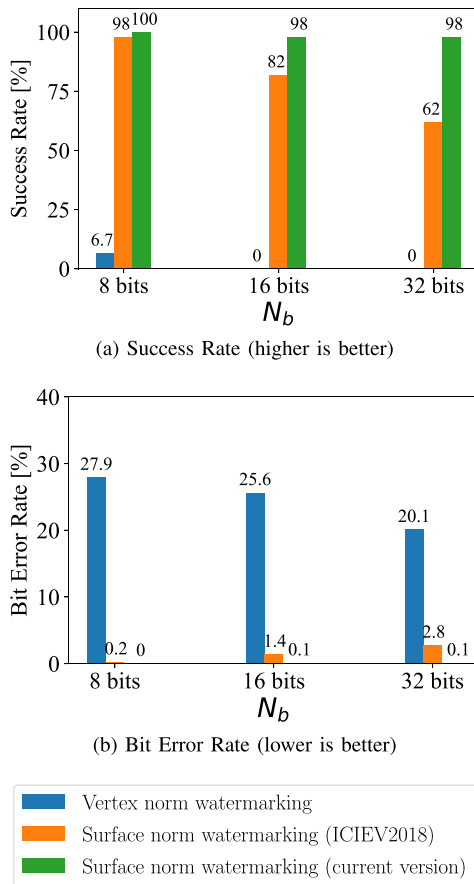


Fig. 20. Comparison between the original vertex norm watermarking method [13], [14], our previous work on surface norm watermarking [12], and the proposed method.

#### H. Comparison to the Previous Work

To evaluate the improvement made by this paper, we compared our method with our previous work [12] and with the original vertex norm watermarking [13], [14]. To obtain reliable results, we evaluated with a printing simulation using the recommended parameters on the complete dataset with 5 different watermarks per model. Figure 20(a) shows the success rate and Fig. 20(b) shows the bit error rate. As expected, the original method based on the vertex norms gives poor results for print/scan because it is not made to be resistant to resampling. Our previous work gives relatively good results at low payload, but lacks robustness when the number of encoded bits increases. Our current method provides a higher success rate and a lower bit error rate, making it superior to the two other methods in a 3D printing context.

#### VI. DISCUSSION AND FUTURE WORK

We presented a new blind watermarking method with low visibility for 3D printed objects, robust to the printing and scanning process, and compatible with most printing and scanning technologies.

We addressed the limitations that prevented using the original vertex norm watermarking method in a 3D printing context. We

made it resistant to resampling by computing the histogram over the entire surface instead of for a discrete set of vertices, and by computing the center of gravity with a moment-based method invariant to resampling. We introduced angular subdivision to prevent the bins from becoming too thin in the radial direction, making them less sensitive to local distortion and misalignment. Finally, we introduced a quantization for the min-max norms to prevent local printing artifacts from misaligning the histogram.

We demonstrated that our method can be applied successfully both in simulation and with actual print/scan situations to multiple objects from a public database encoding up to 32 bits. We also demonstrated that a watermarked object could be scanned and reprinted such that the watermark could still be extracted from the reprinted model.

Our main limitation is the low tolerance for center position misalignment. With our default parameters for printing ( $N_r = 48$ ,  $N_a = 16$ ), the average tolerance is about  $0.2 \sim 0.3$ mm, which is larger than the resolution of most 3D printers (Standard FDM printers have a precision of approximately 0.1 mm, and other technologies such as SLA are even higher precision), but not by a large margin. In future studies, we will attempt to find a method to resynchronize it via known markers or similar methods to achieve a higher tolerance.

We were able to reach the MRMS threshold, but not the MSDM threshold proposed in the mesh watermarking benchmark [11]. However, the MSDM metric, including the threshold chosen in the benchmark, was designed for mesh and not for a 3D printing context. We mainly used these metrics to influence our choice of optimal parameters, making a trade-off between robustness and visibility. In practice, the watermark had low visibility on our printed objects except on flat surfaces such as in ‘Ramesses,’ ‘casting,’ and ‘crank,’ where the artifacts were easier to see.

#### REFERENCES

- [1] H. Lipson and M. Kurman, *Fabricated: The new world 3D printing*: John Wiley & Sons, 2013.
- [2] B. Macq, P. R. Alface, and M. Montanola, “Applicability of watermarking for intellectual property rights protection in a 3d printing scenario,” in *Proc. 20th Int. Conf. 3D Web Technol.*, ACM, 2015, pp. 89–95.
- [3] F. P. Beekhof, S. Voloshynovskiy, O. Koval, R. Villan, and E. Topak, “Document forensics based on steganographic anti-counterfeiting markings and mobile architectures,” in *Proc. 1st Int. Conf. Forensic Appl. Techn. Telecommun., Inf., Multimedia Workshop*. Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 2008, p. 28.
- [4] “Electronic frontier foundation. is your printer spying on you?” [Online]. Available: <http://www.eff.org/Privacy/printers/>, accessed: 2019-01-21.
- [5] G. Walther, “Printing insecurity? the security implications of 3d-printing of weapons,” *Sci. Eng. Ethics*, vol. 21, no. 6, pp. 1435–1445, 2015.
- [6] Z. Li, A. S. Rathore, C. Song, S. Wei, Y. Wang, and W. Xu, “Printracker: Fingerprinting 3d printers using commodity scanners,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, ACM, 2018, pp. 1306–1323.
- [7] L. Zhekun, R. Gadh, and B. Prabhu, “Applications of RFID technology and smart parts in manufacturing,” in *Proc. DETC*, vol. 4. Citeseer, 2004, pp. 1–7.
- [8] D. Wu, D. W. Rosen, L. Wang, and D. Schaefer, “Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation,” *Comput.-Aided Des.*, vol. 59, pp. 1–14, 2015.
- [9] Y. Wu, F. He, D. Zhang, and X. Li, “Service-oriented feature-based data exchange for cloud-based design and manufacturing,” *IEEE Trans. services comput.*, vol. 11, no. 2, pp. 341–353, Mar.–Apr. 2015.

- [10] J. Nieves, I. Ruiz-Agundez, and P. G. Bringas, "Recognizing banknote patterns for protecting economic transactions," in *Proc. Database Expert Syst. Appl., Workshop.*, 2010, pp. 247–249.
- [11] K. Wang, G. Lavoué, F. Denis, A. Baskurt, and X. He, "A benchmark for 3d mesh watermarking," in *Proc. Shape Model. Int. Conf.*, 2010, pp. 231–235.
- [12] A. Delmotte, K. Tanaka, H. Kubo, T. Funatomi, and Y. Mukaigawa, "Blind watermarking for 3-d printed objects using surface norm distribution," in *Proc. Joint 7th Int. Conf. Inform., Electron. Vis., 2nd Int. Conf. Imag., Vis. Pattern Recognit.*, 2018, pp. 282–288.
- [13] J.-W. Cho, M.-S. Kim, R. Prost, H.-Y. Chung, and H.-Y. Jung, "Robust watermarking on polygonal meshes using distribution of vertex norms," in *Proc. Int. Workshop Digital Watermarking*, Springer, 2004, pp. 283–293.
- [14] J.-W. Cho, R. Prost, and H.-Y. Jung, "An oblivious watermarking for 3-d polygonal meshes using distribution of vertex norms," *IEEE Trans. Signal Process.*, vol. 55, no. 1, pp. 142–155, Jan. 2007.
- [15] X. Zhu, J. Ding, H. Dong, K. Hu, and X. Zhang, "Normalized correlation-based quantization modulation for robust watermarking," *IEEE Trans. Multimedia*, vol. 16, no. 7, pp. 1888–1904, Nov. 2014.
- [16] W. Zhang, H. Wang, D. Hou, and N. Yu, "Reversible data hiding in encrypted images by reversible image transformation," *IEEE Trans. multimedia*, vol. 18, no. 8, pp. 1469–1479, Aug. 2016.
- [17] R. Kazemi, F. Pérez-González, M. A. Akhaee, and F. Behnia, "Data hiding robust to mobile communication vocoders," *IEEE Trans. Multimedia*, vol. 18, no. 12, pp. 2345–2357, Dec. 2016.
- [18] T. Stütz, F. Atrousseau, and A. Uhl, "Non-blind structure-preserving substitution watermarking of h. 264/AVLC inter-frames," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1337–1349, Aug. 2014.
- [19] J. M. Konstantinides, A. Mademlis, P. Daras, P. A. Mitkas, and M. G. Strintzis, "Blind robust 3-d mesh watermarking based on oblate spheroidal harmonics," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 23–38, Jan. 2008.
- [20] R. Jiang, H. Zhou, W. Zhang, and N. Yu, "Reversible data hiding in encrypted three-dimensional mesh models," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 55–67, Jan. 2017.
- [21] K. Wang, G. Lavoué, F. Denis, and A. Baskurt, "A comprehensive survey on three-dimensional mesh watermarking," *IEEE Trans. Multimedia*, vol. 10, no. 8, pp. 1513–1527, Dec. 2008.
- [22] N. Medimegh, S. Belaid, and N. Werghi, "A survey of the 3d triangular mesh watermarking techniques," *Int. J. Multimedia*, vol. 1, no. 1, 2015.
- [23] J.-U. Hou, D. Kim, W.-H. Ahn, and H.-K. Lee, "Copyright protections of digital content in the age of 3d printer: Emerging issues and survey," *IEEE Access*, vol. 6, pp. 44 082–44 093, 2018.
- [24] S. Yamazaki, S. Kagami, and M. Mochimaru, "Extracting watermark from 3d prints," in *Proc. Pattern Recognit., 22nd Int. Conf.*, 2014, pp. 4576–4581.
- [25] J.-U. Hou, D.-G. Kim, S. Choi, and H.-K. Lee, "3d print-scan resilient watermarking using a histogram-based circular shift coding structure," in *Proc. 3rd ACM Workshop Inf Hiding Multimedia Secur.*, 2015, pp. 115–121.
- [26] S.-H. Lee, T.-S. Kim, B.-J. Kim, S.-G. Kwon, K.-R. Kwon, and K.-I. Lee, "3d polygonal meshes watermarking using normal vector distributions," in *Proc. Int. Conf. Multimedia Expo., Cat. No. 03TH8698*, vol. 3, 2003, pp. III–105.
- [27] J.-W. Lee, S.-H. Lee, K.-R. Kwon, and K.-I. Lee, "Complex egi based 3d-mesh watermarking," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. 88, no. 6, pp. 1512–1519, 2005.
- [28] Y. Liu, B. Prabhakaran, and X. Guo, "A robust spectral approach for blind watermarking of manifold surfaces," in *Proc. 10th ACM workshop Multimedia Secur.*, 2008, pp. 43–52.
- [29] Z. Karni and C. Gotsman, "Spectral compression of mesh geometry," in *Proc. 27th Annu. Conf. Comput. Graphics Interactive Techn.*, 2000, pp. 279–286.
- [30] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, "A frequency-domain approach to watermarking 3d shapes," in *Proc. Comput. Graphics Forum*, vol. 21, no. 3. Wiley Online Library, 2002, pp. 373–382.
- [31] Y. Liu, B. Prabhakaran, and X. Guo, "Spectral watermarking for parameterized surfaces," *IEEE Trans. Inf. Forensics Secur.*, vol. 7, no. 5, pp. 1459–1471, Oct. 2012.
- [32] A. Kumar, N. P. Goel, and M. Hemani, "Method and apparatus for storing and retrieving data embedded into the surface of a 3d printed object," Jul. 26 2016, uS Patent 9,400,910.
- [33] X. Zhang, Q. Wang, and I. Ivrisimtzis, "Single image watermark retrieval from 3d printed surfaces via convolutional neural networks," *Eurographics Association*, Sep. 2018.
- [34] C. Harrison, R. Xiao, and S. Hudson, "Acoustic barcodes: passive, durable and inexpensive notched identification tags," in *Proc. 25th Annu. ACM Symp. User Interface Software Technol.*, 2012, pp. 563–568.
- [35] M. Molitch-Hou, "The future of hp's multi jet fusion 3d printing," [Online] Available: <https://www.engineering.com/3DPrinting/3DPrintingArticles/ArticleID/122%2098/The-Future-of-HPs-Multi-Jet-Fusion-3D-Printing.aspx>, Jun. 2016, Accessed: 2019-05-21.
- [36] B. Yusuf, "Digitally augmented additive manufacturing parts from rize," [Online] Available: <https://all3dp.com/digitally-augmented-additive-manufacturing-parts-fro%20m-rize>, Apr. 2018, Accessed: 2019-04-24.
- [37] H. T. Maia, D. Li, Y. Yang, and C. Zheng, "Layercode: Optical barcodes for 3d printed shapes," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 112:1–112:14, Jul. 2019. [Online]. Available: <http://doi.acm.org/10.1145/3306346.3322960>
- [38] A. Delmotte, K. Tanaka, H. Kubo, T. Funatomi, and Y. Mukaigawa, "Blind watermarking for 3d printed objects by locally modifying layer thickness," *IEEE Trans. Multimedia*, 2019.
- [39] J. Y. S. Wee, C. I. Byatte, A. D. G. Rhoades, and D. L. McNeight, "Objets de vertu," Dec. 10 2015, U.S. Patent App. 14/485,880.
- [40] J. Y. S. Wee, C. I. Byatte, A. D. G. Rhoades, and D. L. McNeight, "Product authentication," Sep. 3 2015, U.S. Patent App. 14/250,533.
- [41] J. Voris, B. F. Christen, J. Alted, and D. W. Crawford, "Three dimensional (3d) printed objects with embedded identification (id) elements," May 23 2017, U.S. Patent 9,656,428.
- [42] K. D. Willis and A. D. Wilson, "Infrastructs: fabricating information inside physical objects for imaging in the terahertz region," *ACM Trans. Graph. (TOG)*, vol. 32, no. 4, p. 138, 2013.
- [43] A. Okada, P. Silapasuphakornwong, M. Suzuki, H. Torii, Y. Takashima, and K. Uehira, "Non-destructively reading out information embedded inside real objects by using far-infrared light," in *Proc. Appl. Digital Image Process. XXXVIII*, vol. 9599. International Society for Optics and Photonics, 2015, p. 95992V.
- [44] D. Li, A. S. Nair, S. K. Nayar, and C. Zheng, "Aircode: Unobtrusive physical tags for digital fabrication," in *Proc. 30th Annu. ACM Symp. User Interface Softw. Technol.*, 2017, pp. 449–460.
- [45] J.-U. Hou, D.-G. Kim, and H.-K. Lee, "Blind 3d mesh watermarking for 3d printed model by analyzing layering artifact," *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 11, pp. 2712–2725, Nov. 2017.
- [46] R. Hu, P. Rondao-Alface, and B. Macq, "Constrained optimisation of 3d polygonal mesh watermarking by quadratic programming," in *Proc. Acoust., Speech Signal Process. 2009, IEEE Int. Conf.*, pp. 1501–1504.
- [47] C. Zhang and T. Chen, "Efficient feature extraction for 2d/3d objects in mesh representation," in *Proc. Image Process., 2001 Int. Conf.*, vol. 3., 2001, pp. 935–938.
- [48] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: An open-source mesh processing tool," in *Proc. Eurographics Italian Chapter Conf.*, vol. 2008, 2008, pp. 129–136.
- [49] G. Lavoué, E. D. Gelasca, F. Dupont, A. Baskurt, and T. Ebrahimi, "Perceptually driven 3d distance metrics with application to watermarking," in *Proc. Appl. Digital Image Process. XXIX*, vol. 6312. International Society for Optics and Photonics, 2006, p. 63120L.
- [50] G. Hodgson, A. Ranellucci, and J. Moe, "Slic3r manual: Flow math," [Online] <http://manual.slic3r.org/advanced/flow-math>, 2011.
- [51] G. P. Greeff and M. Schilling, "Closed loop control of slippage during filament transport in molten material extrusion," *Additive Manuf.*, vol. 14, pp. 31–38, 2017.



**Arnaud Delmotte** received the M.S. degree in civil engineering in informatics from the University of Louvain, Belgium, in 2013, and the Ph.D. degree in computer science from the Nara Institute of Science and Technology, Japan, in 2020. His research interests include watermarking, 3D printing, and computer vision.



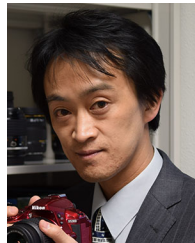
**Kenichiro Tanaka** (Member, IEEE) received the M.S. and Ph.D. degrees in CS from Osaka University, in 2014 and 2017, respectively. In April 2017, he joined the Nara Institute of Science and Technology as an Assistant Professor. His research interests include computer vision and computational photography, imaging, and illumination. He is a member of CVF.



**Takuya Funatomi** (Member, IEEE) received the B.S. degree in engineering, the M.S. and Ph.D. degrees in informatics from the Graduate School of Informatics, Kyoto University, Japan, in 2002, 2004, and 2007, respectively. He is currently an Associate Professor with the Information Science department, Nara Institute of Science and Technology, Japan, since 2015. He was an Assistant Professor with Kyoto University, Japan, from 2007 to 2015, and a Visiting Assistant Professor with Stanford University, USA, in 2014. His research interests include computational photography, 3D model processing, computer vision, computer graphics, and pattern recognition.



**Hiroyuki Kubo** (Member, IEEE) received the M.S. and Ph.D. degrees from Waseda University, in 2008 and 2012, respectively. He is a specially appointed Associate Professor with Tokai University, Japan, since 2020. His research interests include computer graphics and computer vision. He is a member of ACM.



**Yasuhiro Mukaigawa** (Member, IEEE) received the M.E. and Ph.D. degrees from the University of Tsukuba, in 1994 and 1997, respectively. He became a Research Associate with Okayama University, in 1997, an Assistant Professor with the University of Tsukuba, in 2003, an Associate Professor with Osaka University, in 2004, and a Professor with the Nara Institute of Science and Technology, in 2014. His current research interests include photometric analysis and computational photography.