This WACV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Towards Secure and Usable 3D Assets: A Novel Framework for Automatic Visible Watermarking

Gursimran Singh, Tianxi Hu*, Mohammad Akbari*, Qiang Tang, Yong Zhang
Huawei Technologies Canada Co. Ltd.
{gursimran.singh1, cindy.hu1, mohammad.akbari, qiang.tang, yong.zhang3}@huawei.com

Abstract

3D models, particularly AI-generated ones, have witnessed a recent surge across various industries such as entertainment. Hence, there is an alarming need to protect the intellectual property and avoid the misuse of these valuable assets. As a viable solution to address these concerns, we rigorously define the novel task of automated 3D visible watermarking in terms of two competing aspects: watermark quality and asset utility. Moreover, we propose a method of embedding visible watermarks that automatically determines the right location, orientation, and number of watermarks to be placed on arbitrary 3D assets for high watermark quality and asset utility. Our method is based on a novel rigid-body optimization that uses back-propagation to automatically learn transforms for ideal watermark placement. In addition, we propose a novel curvature-matching method for fusing the watermark into the 3D model that further improves readability and security. Finally, we provide a detailed experimental analysis on two benchmark 3D datasets validating the superior performance of our approach in comparison to baselines. Code and demo are available here[1].

1. Introduction

The increasing demand for 3D assets across industries like entertainment [14, 19], augmented reality, and virtual reality [20,28] has driven the adoption of efficient, and scalable methods for creating and distributing content. Generative AI (GenAI) has revolutionized automated 3D content creation, while commercial marketplaces have enhanced distribution networks. This evolution necessitates robust mechanisms to prevent misuse, validate ownership, and protect intellectual property (IP) [26].

Governments and law enforcement agencies are concerned about potential misuse by malicious actors who may exploit automated AI tools to produce controversial 3D content at scale. Such content could be used for spreading misinformation, influencing public opinion, or provoking social unrest. In response, governments (USA [11], China [1], and Europe [16]) are exploring regulations that would mandate GenAI services to embed and publicly disclose the origin of their generated content. In addition, in 3D data marketplaces, sellers must present their 3D assets for potential customers to preview using built-in 3D viewers. However, this practice can be exploited by malicious individuals who download these assets under the guise of previewing them, resulting in significant financial losses for the merchants. Therefore, safeguarding the intellectual property (IP) of these valuable assets within 3D data marketplaces is crucial to prevent unauthorized distribution and ensure fair compensation for the creators.

Digital watermarking is a key technology for copyright protection, source tracking, and authentication [22, 23, 25]. The majority of existing watermarking work in 3D focuses on invisible methods [3, 4, 27, 29, 31]. However, such methods have several drawbacks that limit their practicality in the scenarios discussed above. For instance, in the regulation scenario, it is essential for the general public to identify the asset's origin visually, without requiring specialized tools or knowledge [1, 11]. In contrast, invisible methods rely on specific, often publicly inaccessible, extractors that complicates watermark detection for non-experts. Furthermore, when 3D models are employed in downstream applications like video games or animations, the original watermarked assets become inaccessible for extraction. Consequently, extraction must rely on analyzing 2D visuals that undergo significant alterations, such as changes in lighting and texture, which can hinder successful extraction [32]. Lastly, invisible watermarks, designed to be hidden and imperceptible, do not sufficiently deter unauthorized use in a preemptive manner. Instead, they are more suited as a remedy after the alleged infringement has already happened. In the digital marketplace, copyright infringement

*Equal contribution.

[1] https://developer.huaweicloud.com/develop/aigallery/notebook/detail?id=15adbaaa-2583-4ec3-804a-61c29f001e03

can be easily committed through the replication of digital assets. However, identifying and proving such infringement is challenging, leading to complex and costly legal actions.

In this work, we propose the novel task of automated 3D visible watermarking as a viable alternative. The objective is to embed a copyright mark, such as a visible logo or text message, across different areas of the 3D model surface. These visible watermarks are designed to overcome the limitations associated with invisible methods discussed previously. Specifically, in regulatory contexts, smaller visible watermarks can be strategically positioned in less intrusive areas of 3D models. This approach aims to maintain the model's functionality while using watermarks as stamps to track provenance information. Thus, the general public can readily interpret visible watermarks to discern whether the content is AI-generated or created by humans. On the other hand, for merchandising scenarios, we can strategically place large and prominent visible watermarks at various locations on the asset that act as a strong deterrent against unauthorized use. This encourages interested customers to purchase the unwatermarked model, ensuring fair compensation for the asset owner. Hence, this approach offers proactive protection of intellectual property compared to relying solely on legal action after misuse.

Automated 3D visible watermarking is a challenging task due to several reasons. First, the algorithm needs to automatically identify the most suitable locations (specific coordinates) on the 3D model surface to embed watermarks. In other words, the algorithm must choose locations that boost the visibility of watermarks for high security, while preserving the most salient and prominent features of the original model. Second, the algorithm must guarantee that the entire watermark is well-aligned along the model's surface curvature with no component floating, as such parts can be easily removed using an isolated parts detection algorithm, thereby compromising the security of the watermarks. Finally, the algorithm must fuse the watermarks into the 3D geometry rather than merely inscribing them into the model texture. This makes it an irreversible process where the watermarks are very hard and expensive to remove.

To address these challenges, we introduce an end-to-end pipeline for automatically embedding visible watermarks on arbitrary 3D models. Our solution proposes a novel gradient-based optimization method that generates a large number of candidate watermarks placed throughout the surface of the model. Having generated these diverse candidates, we define a filtering-based algorithm to pick the final set of watermarks based on the specific utility and security requirements. Finally, we propose a novel Boolean-based mesh merging to fuse watermarks into the geometry while matching the local curvature of the underlying surfaces. The major contributions of this work are as follows:

- We define the novel task of automatic 3D visible watermarking, where the main goal is to determine the number, orientation, and location of watermarks to achieve the best watermark quality and best asset quality. We also provide rigorous definitions for various aspects of both watermark quality and asset quality.

- We propose a novel end-to-end pipeline as a solution for the task of automated 3D visible watermarking. Our solution is based on a gradient-based optimization to determine the location and rotation transforms for the best orientation. Additionally, we propose a novel curve-matching fusion for enforcing the watermarks to follow the local curvature of the underlying surface.

- We propose a holistic evaluation benchmark for 3D visible watermarking for future research. We propose practical metrics for various aspects of watermark and asset quality. Further, we provide detailed experiments on three 3D datasets to demonstrate the superiority of our approach in comparison to the baselines.

2. Related Works

Watermarking 3D models is a problem of great interest [2–4, 6, 31]. While 3D invisible watermarking [3, 4, 27, 29, 31] is well-established, the field of 3D visible watermarking remains in its infancy. Traditional approaches [2, 5, 21] use edge subdivisions to carve watermark characters on a smooth surface of the mesh. These approaches give a visual effect of watermark characters, however, they are designed to be only viewed using mesh-editing software that can display edge information. Hence, they are not visible to the naked eye in rendering mode or after printing the 3D model. Another limitation of these methods is that they do not support 3D formats without any edge information (e.g., voxel format). Recently, Li et al. [13] proposed an approach to embed a single 3D watermark on a model using mesh Boolean operations, a technique similar to ours. However, their method has several limitations. First, they do not provide a framework for automatic identification of the locations to put watermarks on the surface of the model. Instead, they only provide some guidelines for humans to select locations manually. Second, their algorithm does not support fixing the orientation of watermarks, which is critical to ensure security against removal attacks. Finally, the specific Boolean merging operations proposed in this work do not match the curvature of the underlying surface, leading to poor visual quality of results. On the other hand, our method automatically selects and places watermarks on the surface of the 3D mesh with a focus on both watermark and asset quality.

3. Problem Definition

At a high level, the task of 3D visible watermarking is to automatically generate the watermarked (output) model by embedding multiple watermarks on the target (input) model. Formally, the inputs to the task include: 1) *Target 3D mesh* $M(V, F)$ consisting of a set of vertices $V = \{v_i | v_i \in \mathbb{R}^3\}$, a set of faces $F = \{(v_i, v_j, v_k) | v_i, v_j, v_k \in V\}$; 2) watermark text; and 3) algorithm parameters such as font, thickness, and size of watermarks.

The output is *a 3D mesh* $M'(V', F')$ with a variable number of watermarks, denoted by H_f, embedded into its surface at multiple locations. The new set of vertices V' and faces F' represent the modified geometry where the watermarks are an inseparable part of the watermarked mesh M'. Additionally, let's denote the sub mesh corresponding to the i^{th} embedded watermark as $W_i \subset M'$ and an assumed minimum volume bounding box around it by B_i.

The central problem in 3D visible watermarking is to determine the number, location, and orientation of the watermarks to be placed on the target model's surface. For instance, having too many watermarks can obscure crucial details of the model, making it unusable for downstream applications. Conversely, too few watermarks may compromise security, allowing adversaries to exploit views without watermarks to create 2D renderings illegitimately. Similarly, the location and orientation of the watermarks have security-vs-utility consequences that need to be kept in mind during the watermarking process. To make it concrete, we group these security and utility consequences into two main groups - *watermark quality* and *asset utility*, and discuss them in detail in the following sub-sections.

3.1. Watermark Quality

Watermark quality encompasses various aspects of the security of the watermarked asset. In order to have high security, the placed watermarks must be 1) hard to remove by an adversary, 2) easily readable by a human eye, and 3) viewable from multiple camera angles. To make the problem concrete, we define these individual aspects into quantifiable and measurable metrics called watermark placement and watermark visibility.

Watermark Placement: The principle is that watermarks should be precisely aligned on the surface of the target mesh to enhance security, readability, and visual quality. Misaligned watermarks, where characters are embedded within the mesh surface, are undesirable for several reasons. Firstly, isolated characters can be easily detected and removed by automated algorithms, compromising the security of the watermark. Secondly, embedded parts make it difficult to fully read the watermark, affecting its readability. Lastly, watermarks that do not conform to the mesh surface appear visually less appealing to human observers.

Definition 1. *Watermark placement is quantified as the average proportion of the intersection area between the original mesh M and each watermark mesh W^i from the set of all watermarks $\{W^i\}_{i=1}^{H_f}$. This is computed as:*

$$\mathcal{P}(M, M') = \frac{1}{H_f} \sum_{i=1}^{H_f} \max \left\{ \frac{\mathcal{A}_{\hat{n}^i}(M \cap W^i)}{\mathcal{A}_{\hat{n}^i}(W^i)}, 1 \right\} \quad (1)$$

where $\mathcal{A}_{\hat{n}}(w)$ represents the projected area of a mesh w towards a normal vector $\hat{n}$. $W^i \subset M'$ is the i^{th} watermark mesh and $\hat{n}^i$ is the normal vector facing the front side of the corresponding watermark mesh.

Watermark Visibility: It quantifies whether watermarks are clearly visible and readable from various camera angles around the watermarked mesh, which is crucial for enhancing security, as higher visibility prevents adversaries from obtaining renders (2D shots) of the asset without watermarks. Additionally, improved visibility ensures better readability, enabling investigators to easily identify and verify the presence of the watermark from multiple angles.

Definition 2. *Consider a set of camera views $\{c^t\}_{t=1}^{T}$ obtained by randomly rotating a camera around the watermarked object. Assume $\mathcal{K}_{\mathcal{V}}(M', c^t)$ as a kernel, where it equals 1 if a watermark on M' is visible in view c^t to a human, and 0 otherwise. For a large number T of views, we define the watermark visibility $V(M')$ as the proportion of views where at least one watermark is visible:*

$$\mathcal{V}(M') = \frac{1}{T} \sum_{t=1}^{T} \mathcal{K}_{\mathcal{V}}(M', c^t) \quad (2)$$

3.2. Asset Utility

Asset utility encompasses various factors that affect the usefulness of the watermarked asset across different downstream applications. Given the subjective nature and complexity of defining downstream utility, our focus is primarily on assessing any potential degradation caused by watermarking. Specifically, we evaluate how the watermarking process impacts the asset in terms of geometry, saliency, and semantic integrity. For example, adding a greater number or larger size of watermarks to an asset results in more significant degradation. Similarly, placing watermarks in highly noticeable and salient areas of the asset can diminish its utility more than choosing less conspicuous, flat areas. Accordingly, we define specific criteria for assessing asset utility: geometry similarity, saliency preservation, and semantic preservation.

Geometry Similarity: This metric quantifies alterations in the appearance-level 3D geometry of the watermarked asset M' relative to the original asset M. It focuses on detecting changes such as surface geometry modifications, variations in local curvature, and occlusions of surface features resulting from the watermarking process.

Definition 3. *Let $K_G(M', M, dr)$ denote a kernel quantifying the local 3D geometry similarity between a*

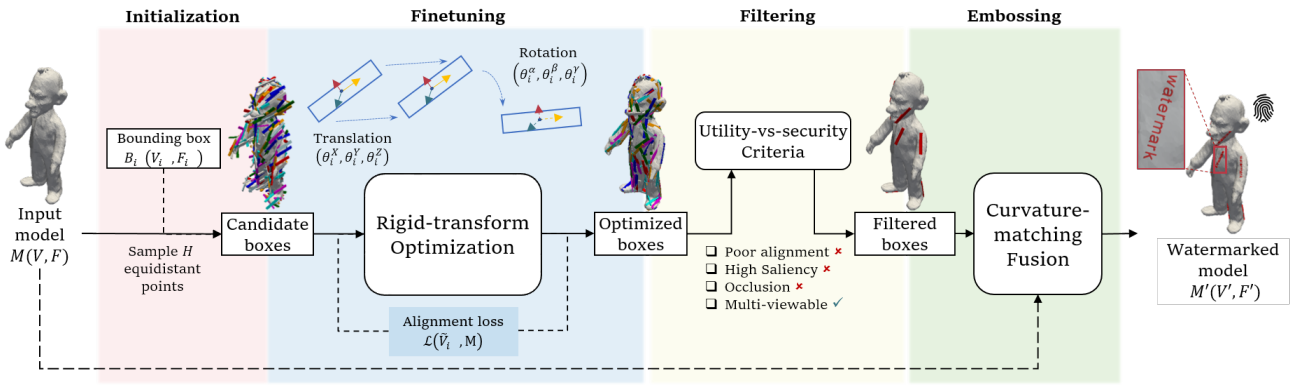


Figure 1. The overall framework of our proposed automatic 3D visible watermarking.

small region dr of the target mesh M and the corresponding region of the watermarked mesh M'. The geometry similarity is defined as:

$$\mathcal{G}(M,M') = \int_r \mathcal{K}_{\mathcal{G}}(M,M',dr)\cdot dr \tag{3}$$

Saliency Preservation: This criterion quantifies whether the salient and highly noticeable features of the original mesh are preserved during watermarking. For instance, in the context of a cat object, critical features like the face, ears, and paws must be preserved to ensure the continued usefulness of the model for downstream applications.

Definition 4. *Let $\mathcal{K}_{\mathcal{S}}(r)$ denote a kernel for computing an estimate of the average saliency of a local region r in the original model M. Using a threshold τ_s determined by Otsu's method [30], we define the saliency retention as:*

$$\mathcal{S}(M,M') = \frac{1}{H_f}\sum_{i=1}^{H_f}\mathbb{I}(\mathcal{K}_{\mathcal{S}}(M\cap B^i) < \tau_s) \tag{4}$$

where $\mathbb{I}(c)$ is an indicator function whose value is 1 if c is true, otherwise 0.

Semantics Preservation: This criterion addresses the potential degradation in high-level semantic concepts caused by watermarking. To maintain high semantics preservation, watermarks should avoid locations that could alter or obscure the semantic understanding of the object. For example, adding numerous watermarks to the face of an animal might make it challenging to discern whether it semantically represents a cat or a dog.

Definition 5. *Let $\{c_o^t\}_{t=1}^T$ and $\{c_w^t\}_{t=1}^T$ denote sets of corresponding camera views obtained by repeatedly and randomly rotating a camera around the original and watermarked objects, respectively. Assume $\mathcal{K}_{\mathcal{F}}(c_o^t, c_w^t)$ as a kernel for estimating the semantic similarity between corresponding views c_o^t and c_w^t. For a large value of T, semantics preservation is defined as:*

$$\mathcal{F}(M,M') = \frac{1}{T}\sum_{t=1}^{T}\mathcal{K}_{\mathcal{F}}(c_o^t, c_w^t) \tag{5}$$

3.3. Watermark Quality vs. Asset Utility

The ultimate goal of the 3D visible watermarking task is to obtain high asset utility and watermarking quality. Ideally, the asset utility before and after watermarking should be approximately the same. At the same time, we would like high-quality watermarks that are visible from all angles to the human eye. However, watermarking often involves a trade-off between asset utility and watermark quality. The goal of this work is to improve the fundamental trade-off by improving both aspects of asset utility and watermark quality together.

4. Method

In this section, we outline our proposed solution for automated 3D visible watermarking. Fig. 1 presents a high-level visualization of the entire process. Our pipeline is composed of four main modules: initialization, finetuning, filtering, and embossing. Initially, we generate a large number of candidate boxes, that serve as placeholders for the actual 3D watermarks, on the surface of the target model. Next, the finetuning module adjusts the position and orientation of these boxes to ensure they flow along the surface of the mesh rather than protruding out or into the surface. Then, the filtering module selects the final subset of watermarks with an aim to fulfill the high watermark quality and high asset utility requirements. Finally, the embossing module integrates the 3D text watermark into the target model's surface, ensuring the watermarks conform to the local surface curvature for optimal visual appeal.

4.1. Initialization

We begin by sampling a set of H equidistant points on the surface of the target model. At each sampled point i, we create a 3D rectangular box denoted as $B_i(V_i, F_i)$ and orient it to face along the corresponding surface normal. These boxes collectively serve as placeholders that will later be substituted with 3D text watermarks. For a detailed de-

scription of the initialization algorithm, please consult the supplementary materials.

4.2. Finetuning

The boxes generated in the initialization phase need to be fine-tuned for optimal watermark placement. Specifically, they need to be moved to appropriate locations and rotated to properly align along the surface of the model. Since the required translation and rotation are unknown in advance, we propose a novel rigid-transform optimization approach to estimate the necessary transforms automatically.

To accomplish this, we introduce rotation parameters $(\theta_i^\alpha, \theta_i^\beta, \theta_i^\gamma)$ and translation parameters $(\theta_i^X, \theta_i^Y, \theta_i^Z)$. Here, the rotation parameters specify the angles of rotation, while the translation parameters indicate the displacements along the X, Y, and Z axes, respectively. Using these parameters, we apply translation and rotation operations to each candidate box. For the i^{th} candidate box, the transformed vertices $\tilde{V}_i$ are obtained as follows:

$$\tilde{V}_i = T_{C_i^X, C_i^Y, C_i^Z} \cdot R_{\theta_i^\alpha, \theta_i^\beta, \theta_i^\gamma} \cdot -T_{C_i^X, C_i^Y, C_i^Z} \cdot \bar{V}_i \quad (6)$$

$$\text{where } \bar{V}_i = T_{\theta_i^X, \theta_i^Y, \theta_i^Z} \cdot V_i, \quad (7)$$

where Eq. (7) represents the parameterized translation step of moving the vertices of i^{th} bounding box V_i using the translation matrix $T_{\theta_i^X, \theta_i^Y, \theta_i^Z} \in \mathbb{R}^{3\times3}$. Eq. (6) represents the parameterized rotation of the box around its centroid using the rotation matrix $R_{\theta_i^\alpha, \theta_i^\beta, \theta_i^\gamma} \in \mathbb{R}^{3\times3}$. Note, the translation matrix $-T_{C_i^X, C_i^Y, C_i^Z}$ is required to temporarily move the box at origin (0, 0, 0), which is eventually transported back to its previous place (C_i^X, C_i^Y, C_i^Z) determined by Eq. (7). Moving the box temporarily to the origin is required to guarantee that the box is rotating around its centroid (C_i^X, C_i^Y, C_i^Z) and not an arbitrary point.

Finally, we define the optimization objective as follows:

$$\underset{\{\theta_i^\alpha, \theta_i^\beta, \theta_i^\gamma, \theta_i^X, \theta_i^Y, \theta_i^Z\}_{i=1}^H}{\arg\min} \frac{1}{H} \sum_{i=1}^{H} \mathcal{L}(\tilde{V}_i, M), \quad (8)$$

where $\mathcal{L}$ is the loss function defined in the next section.

Loss Function. In essence, it provides an assessment of the discrepancy in alignment between the box $\tilde{V}_i$ and the mesh surface M. Ideally, the mesh surface should intersect the box at its midpoint, effectively halving it. This alignment guarantees that the characters of the embedded watermark remain parallel to the surface, ensuring optimal readability and security.

Let t_1, t_2, t_3, t_4 denote consecutive vertices on the front face of the box, and b_1, b_2, b_3, b_4 denote corresponding vertices on the bottom face. This gives us midpoints $m_1 = \frac{b_1+t_1}{2}, m_2 = \frac{b_2+t_2}{2}, m_3 = \frac{b_3+t_3}{2}$, and $m_4 = \frac{b_4+t_4}{2}$ on the lateral faces of the box. Then, we simply sample equidistant points along the line segments defined by points $(m_1, m_2), (m_2, m_3), (m_3, m_4)$, and (m_4, m_1). We denote all sampled points, including the midpoints, by the set $\{s_i^j\}_{j=1}^J$, where J is the total number of points per box. Using this set, we define the loss as the distance between these sampled points and the surface of the mesh:

$$\mathcal{L}(\tilde{V}_i, M) = \frac{1}{J} \sum_{j=1}^{J} \mathcal{D}(s_i^j, M), \quad (9)$$

where $\mathcal{D}$ denotes the built-in differential loss in PyTorch3D [24] used to compute distances between point clouds and meshes. Intuitively, minimizing this loss ensures that all points $\{s_i^j\}_{j=1}^J$ lie on the mesh surface. This alignment is achieved when the mesh surface passes through the middle of the box, parallel to both the front and back faces, thereby bisecting the box into halves.

Learning. The optimization objective specified in Eq. (8) is nonlinear and non-convex. We simply use back-propagation to compute gradients of objective in Eq. (9) and obtain the optimized parameters $\{\overset{*}{\theta_i^\alpha}, \overset{*}{\theta_i^\beta}, \overset{*}{\theta_i^\gamma}\}_{i=1}^H$ and $\{\overset{*}{\theta_i^X}, \overset{*}{\theta_i^Y}, \overset{*}{\theta_i^Z}\}_{i=1}^H$ by minimizing the objective in Eq. (9). This process yields the refined bounding-box vertices $\{V_i^*\}_{i=1}^H$ and their corresponding meshes $\{B_i^*(V_i^*, F_i)\}_{i=1}^H$.

Due to the non-convex nature of the optimization, it is possible to obtain sub-optimal solutions, resulting in poorly aligned boxes. To address this, we start with a large number H of candidate bounding boxes distributed across the mesh's surface. Then, we apply heuristics to eliminate sub-optimal solutions, as detailed in the following section.

4.3. Filtering

This module aims to select a subset of size $H_f \ll H$ from the optimized boxes obtained in the previous step. Guided by the utility-vs-security criteria outlined in Sec. 3, the goal is to ensure robust watermark security while preserving the utility of the watermarked asset with minimal degradation. Specifically, we do this by incrementally pruning sub-optimal, unnecessary, and redundant watermarks using a series of filtering steps.

We start by rejecting sub-optimal boxes that are poorly oriented (high loss) or are placed on highly salient regions [5] of the mesh. Then, we remove boxes that have poor visibility due to potential occlusion by sub-parts of the original mesh using a ray casting approach. Next, we strategically choose the minimal set of watermarks to meet security criteria. Specifically, the model is divided into eight octants by segmenting it with X, Y, and Z planes. Subsequently, we employ a greedy method to choose one box per octant, aiming to maximize the spacing between selected boxes in the solution set. Following this, we search for locations for watermarks using fixed angle increments of 30° around the X and Z axes. New watermarks are added if no existing watermarks are already positioned at that angle. These two steps ensure that the watermarks are viewable from multiple angles for high watermark security. Please refer to the supplement for more details about these steps.

4.4. Embossing

Having obtained the locations and orientations of the final set of H_f candidate boxes, we generate the corresponding 3D-text watermark meshes using a standard text-to-3D algorithm [17]. However, these watermarks remain discrete and disconnected objects (individual 3D characters), which could be identified and removed by a knowledgeable attacker, thereby compromising security. Additionally, these watermarks create a flat textual surface on potentially curved surfaces, which can significantly degrade the visual quality (utility) of the watermarked asset (see Fig. 2).

We propose a novel method for enhancing watermark meshes by aligning them with the local curvature of the underlying surface through a curve-matching fusion. Initially, we compute the intersection between the target model and the 3D-text watermark meshes. Subsequently, we extrude the intersection result at all vertices in the direction of the bounding box normal by a consistent distance. This adjustment results in updated watermark meshes that conform to the local curvature, maintaining a fixed distance from the underlying surface. Finally, we apply Boolean operations such as union and difference, as described in [13], to achieve embossing or debossing effects. For a detailed algorithm, please refer to the supplementary materials.

5. Experiments

In this section, we analyze and compare the performance of the proposed automated 3D visible watermarking method both quantitatively and qualitatively against the baselines.

Baseline Methods. To the best of our knowledge, there is no previous work with the capability of automatically determining the locations of watermarks. Hence, none of the existing baselines is directly applicable to the task of 3D visible watermarking presented in this work. For the sake of comparison, we have re-purposed the most recent and capable baseline Li et al. [13] and assumed random locations on the 3D mesh for placing watermarks in an automated manner. All other methods [2, 5, 21] have significant shortcomings (see Sec. 2) which limit their applicability to the practical scenarios of 3D visible watermarking.

Datasets and Implementation Details. We utilize three 3D datasets for our experiments. The first two datasets consist of 50 models randomly sampled from well-known benchmark datasets: Manifold40 [12] and ObjaVerse [9]. The third dataset comprises 20 textured models obtained from the Meshy [15] text-to-3D generative AI service. Our method dynamically chooses the number of watermarks as determined by the filtering step (Sec. 4.3). For a fair comparison, in all our experiments, we choose the same number of watermarks for the Li et al. baseline. More implementation details, running time analysis, and statistics of the datasets are given in the supplementary materials.

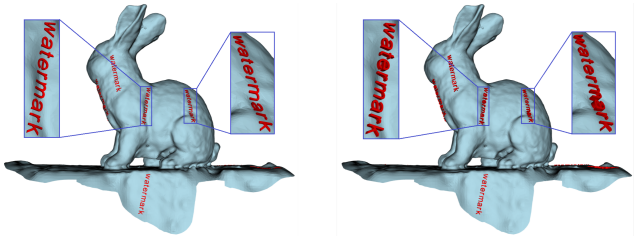


Figure 2. Qualitative analysis of our method with (left) and without (right) curve-matching fusion on an example from Objaverse.

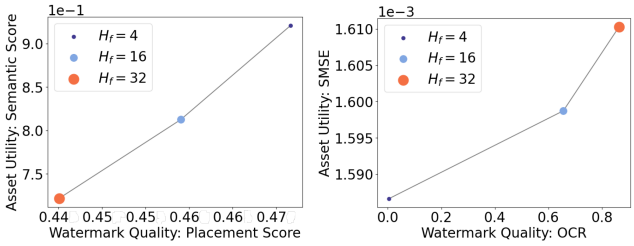


Figure 3. Trade-off results between the watermark quality and asset quality metrics on Manifold40. H_f: number of watermarks.

Evaluation Metrics. For watermark quality, we define three metrics, namely *WPS*, *Ray*, and *OCR*, based on the concepts of watermark placement and visibility defined in Section 3. WPS is computed using Eq. (1), except that we approximate the watermark mesh W_i by its oriented bounding box B_i to efficiently compute the intersection area. The Ray and OCR scores are computed by Eq. (2) using a fixed number of camera views T sampled around the X and Z axis. Moreover, for Ray, the kernel $\mathcal{K}_{\mathcal{V}}(M', c^t)$ is implemented via ray casting that approximates viewability by firing multiple rays from the front face of a watermark's bounding box and checks whether all of them can reach the camera uninterrupted. On the other hand, for OCR, we use a standard OCR method [8] to approximate $\mathcal{K}_{\mathcal{V}}(M', c^t)$, that outputs 1 if at least one watermark can be correctly recognized in the 2D render of a certain view c^t.

For asset utility, we define five different scores. The first three, Sampled Mean Squared Error (SMSE), Isolated Parts Error (IPE), and Local Curvature Error (LCE) are based on geometric similarity defined in Eq. (3). SMSE score is similar to the concept of Mean Squared Error (MSE), except that it is between two meshes instead of a vector of points. The IPE metric calculates the difference in total isolated parts between the watermarked mesh M' and the original mesh M. This difference increases when watermarks are poorly aligned on the model surface, causing certain 3D letters to float above or below the surface. The LCE score assesses curvature preservation post-watermarking (discussed in Sec. 4.4). It calculates the variance in distance from the mesh surface to the top of the watermark in watermarked areas. For curve-preserving watermarks, this distance remains constant, resulting in lower variance.

The remaining two scores Saliency Error (SE) and Se-

Dataset	Method	Watermark Quality			Asset Utility				
		WPS ↑	Ray ↑	OCR ↑	SMSE ↓	IPE ↓	LCE ↓	SE ↓	SS ↑
Manifold40	Li et al.	0.280	0.706	0.301	0.006	20.525	0.022	0.102	0.795
	Ours	**0.484**	**0.920**	**0.369**	**0.002**	**0.575**	**0.000**	**0.093**	**0.832**
Objaverse	Li et al.	0.203	0.588	0.201	0.009	19.400	0.038	0.168	0.792
	Ours	**0.420**	**0.788**	**0.355**	**0.002**	**4.820**	**0.002**	**0.021**	**0.811**
Meshy	Li et al.	0.207	0.837	0.362	0.0075	27.75	0.0293	0.146	0.822
	Ours	**0.411**	**0.993**	**0.526**	**0.0017**	**0.35**	**0.0006**	**0.020**	**0.856**

Table 1. Comparison results in terms of different watermark quality and asset quality metrics on Manifold40, ObjaVerse, and Meshy datasets. ↑: higher is better. ↓: lower is better.

mantic Score (SS) are based on definitions Eq. (4) and Eq. (5). Specifically, to implement the saliency kernel $\mathcal{K}_{\mathcal{S}}(w)$, we pick an off-the-shelf algorithm [18] to compute the saliency map and report the average saliency of the sub mesh w as the output. For the semantic kernel $\mathcal{K}_{\mathcal{F}}(c_o^t, c_w^t)$, we compute the cosine similarity between the semantic features of 2D renders for views c_o^t and c_w^t using a pre-trained ResNet50 [10] feature extractor.

5.1. Quantitative Results

The comparison results of our method and the baseline in terms of watermark quality and asset utility aspects are summarized in Tab. 1. Overall, our approach outperforms the baseline on all the metrics by a significant amount. In particular, as indicated by the WPS and Ray scores, the watermarks embedded using our method achieve ≈20% higher surface alignment (WPS) and multi-view visibility (Ray) compared to the baseline. Moreover, our OCR scores are 7%, 15%, and 16% higher for Manifold40, ObjaVerse, and Meshy, respectively, which show the superiority of our method in watermark text readability.

On the other hand, for the asset utility, our method achieves ≈4X lower SMSE error rate on average, showing high geometry similarity between the watermarked and original 3D assets. In addition, we achieve significantly lower IPE and LCE error rates that guarantee very low isolated parts and high curvature preservation after watermarking, respectively. Further, the results also demonstrate our method's superiority for the preservation of the salient features and semantic context of the asset after watermarking.

In order to study the trade-off between the watermark quality and asset utility (discussed in Sec. 3.3), we perform another set of experiments with a different number of watermarks $H_f = \{4, 16, 32\}$. Two trade-off curves including Semantic vs. Placement and SMSE vs. OCR are illustrated in Fig. 3. Each point on the curve represents the results averaged over the models in the Manifold40 dataset. In general, increasing the number of watermarks results in higher watermark quality, but lower asset utility. However, compared to the baseline, our method achieves a significantly improved trade-off. In particular, for the same semantic score of ≈0.80 (given in Tab. 1), our method achieves a placement score of 0.46 which is 18% higher than the baseline. Moreover, our method can achieve an OCR score of 0.60, while providing an SMSE error of 0.0016. On the other hand, the baseline can achieve a much lower OCR score of 0.30 with a much higher SMSE error of 0.006. More trade-off curves are provided in the supplementary materials.

5.2. Qualitative Results

Fig. 4 shows the qualitative results of our method and the baseline. The watermarks are colored red to enhance visibility for the reader. Our method generates watermarks that are better aligned with the surface and visible from multiple views. On the other hand, the baseline produces watermarks that either fly out or are hidden under the surface, resulting in poor visibility. Further, in Fig. 2, a qualitative analysis of our method with and without the proposed curve-matching fusion module is shown. From the enlarged areas, it is clear that including this fusion enforces the watermarks to follow the surface curvature, which provides higher watermark visibility and asset utility. More qualitative results along with analysis are given in the supplementary materials. Moreover, to subjectively analyze the performance of our method compared to the baseline, we conducted a user study, which is summarized in the supplementary materials.

5.3. Ablation Study

We perform an ablation study to analyze the effect of the components of our method including rigid-transform optimization (Sec. 4.2), curvature fusion (Sec. 4.4), and the filtering steps (Sec. 4.3). The corresponding watermark quality and asset utility results on the ObjaVerse dataset are presented in Tab. 2. As summarized in the table, excluding any of the optimization or filtering modules negatively affects both watermark quality and asset utility metrics. Specifically, our method without the optimization stage not only provides approximately 20% lower WPS and Ray scores, but it also reduces the SMSE and LCE error rates by 2× and 7×, respectively. For the filtering steps, the results are impacted less as the optimization module contributes more to the overall performance of the method. It is also seen that excluding the curve-matching fusion module results in a much higher LCE error, which shows the significance of

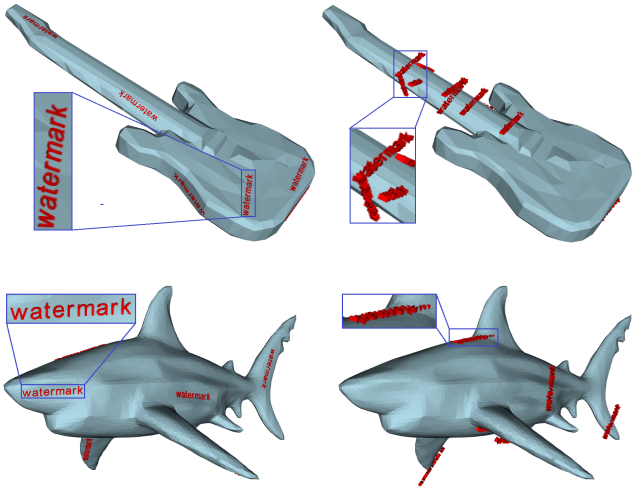


Figure 4. Visual example of 3D models from Manifold40 (top) and Meshy (bottom) watermarked with our method (left) and Li et al. baseline (right). Ours provides better placement quality, readability, and viewability.

this module in preserving the surface curvature of the asset.

Method	Watermark Quality			Asset Utility			
	WPS ↑	Ray ↑	OCR ↑	SMSE ↓	IPE ↓	LCE ↓	SS ↑
Ours	**0.420**	**0.788**	**0.355**	**0.002**	4.820	**0.002**	**0.811**
w/o CF	0.420	0.788	0.294	0.002	4.820	0.021	0.781
w/o Optimize	0.225	0.543	0.232	0.005	**3.800**	0.015	0.786
w/o Filtering	0.287	0.594	0.259	0.002	7.870	0.004	0.800
Li et al.	0.203	0.588	0.201	0.009	19.400	0.038	0.792

Table 2. Ablation over different components of our method on the ObjaVerse dataset. CF: curvature fusion.

5.4. Attack and Robustness Analysis

First, we analyze the robustness of our method in comparison to Li et al. [13] baseline. We consider the cropping and unauthorized removal attacks, which are chosen to reflect strong adversaries. The results are summarized in Tab. 3. In the crop attack, an attacker aims to illegitimately crop a significant part of the model. The attack severely impacts both watermark quality and asset utility. Despite this, our method surpasses the baseline as it applies watermarks at multiple angles, thereby preserving more watermarks and leading to improved watermark quality metrics.

On the other hand, for the unauthorized removal attack, we assume the attacker can remove the vertices and faces belonging to the watermark, probably using manual or automated methods. As shown in Tab. 3, our method preserves the watermark quality that degrades significantly for the baseline. This is because removing watermarks in our method leaves holes in the mesh surface, creating a silhouette through which the watermark can still be read clearly. In contrast, in baseline, parts of the watermark that are not in direct contact can be completely removed, resulting in the watermark being unreadable. Please refer to the supplement for a qualitative comparison.

Lastly, we analyze the performance of our watermarks against the varying strengths of the remeshing attack, which

Attack	Method	Watermark Quality		Asset Utility	
		Ray ↑	OCR ↑	IPE ↓	SS ↑
No Attack	Li et al.	0.706	0.301	20.525	0.795
	Ours	**0.920**	**0.369**	**0.575**	**0.832**
Crop Attack	Li et al.	0.501	0.093	9.650	0.685
	Ours	**0.569**	**0.120**	**4.050**	**0.716**
Removal Attack	Li et al.	0.541	0.211	**20.350**	0.918
	Ours	**0.920**	**0.356**	29.025	**0.927**

Table 3. Quantitative results of our method compared to the baseline against crop and removal attacks.

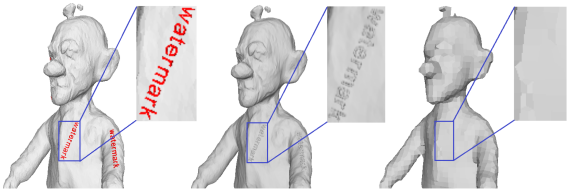


Figure 5. Left shows the original model; middle and right show the results of the remeshing attack with low and high strength.

is based on blender's Remesh Modifier [7]. As depicted in Sec. 5.4, at low attack strength (middle), both asset utility and visible watermarks are reasonably preserved, albeit with loss of texture during remeshing. As the attack strength increases (right), visible watermarks are removed but the asset utility is also significantly impaired, particularly damaging facial features.

5.5. Comparison with Invisible Watermarking

Here, we provide a comparison of performance and robustness with an invisible baseline, Wang et.al. [29]. The performance results are shown in Tab. 4. As expected, the invisible technique performs poorly on the watermark visibility criteria of our benchmark, signaling its inability to meet the demands of the proposed task. Further, we found that this baseline was completely ineffective ($< 50\%$ bit accuracy) against the cropping and remeshing attacks (even low strength), signaling its poor robustness in comparison to the proposed visible method.

Method	Ray ↑	OCR ↑	SMSE ↓	SS ↑
Invisible	0.000	0.000	**7.81** $\times 10^{-9}$	**0.999**
Ours	**0.920**	**0.369**	0.002	0.832

Table 4. Comparison with invisible method, Wang et.al. [29].

6. Conclusion

In this paper, we tackled the novel task of automatically embedding 3D visible watermarks to arbitrary 3D models. We first defined the objectives of the task of 3D visible watermarking in terms of various aspects of watermark and asset quality. Then, we proposed an end-to-end pipeline that uses a gradient-based optimization to achieve high watermark quality and high asset utility. We conducted an extensive set of experiments on two benchmark 3D datasets to demonstrate the effectiveness of our approach. Through our work, we aim to further research in the novel and practical direction of 3D visible watermarking. The limitations of our work are discussed in the supplementary materials.

References

[1] National Technical Committee 260 0n Cybersecurity of Standardization Administration of China. Cybersecurity standard practice guide - methods for content identification of generative ai services.

[2] X.-C An, Rongrong Ni, and Yao Zhao. Visible watermarking for 3D models based on boundary adaptation and mesh subdivision. 34:503–514, Sept. 2016.

[3] O. Benedens. Geometry-based watermarking of 3D models. *IEEE Computer Graphics and Applications*, 19(1):46–55, Jan. 1999.

[4] Adrian G. Bors and Ming Luo. Optimized 3D watermarking for minimal surface distortion. *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 22(5):1822–1835, May 2013.

[5] Jinliang Cao, Zhiwei Niu, Anhong Wang, and Li Liu. Reversible Visible Watermarking Algorithm for 3D Models. 2020.

[6] François Cayre, Patrice Rondao Alface, Francis Schmitt, Benoit Macq, and Henri Maître. Application of spectral decomposition to compression and watermarking of 3D triangle mesh geometry. *Sig. Proc.: Image Comm.*, 18:309–319, Apr. 2003.

[7] Blender contributers. Blender 4.1 manual: Remesh modifier.

[8] Keras OCR contributers. A packaged and flexible version of the craft text detector and keras crnn recognition model.

[9] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A Universe of Annotated 3D Objects, Dec. 2022. arXiv:2212.08051 [cs].

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, Dec. 2015. arXiv:1512.03385 [cs].

[11] The White House. Executive Order on the Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence, Oct. 2023.

[12] Shi-Min Hu, Zheng-Ning Liu, Meng-Hao Guo, Jun-Xiong Cai, Jiahui Huang, Tai-Jiang Mu, and Ralph R. Martin. Subdivision-Based Mesh Convolution Networks. *ACM Transactions on Graphics*, 41(3):1–16, June 2022. arXiv:2106.02285 [cs].

[13] An-Bo Li, Hao Chen, and Xian-Li Xie. Visible watermarking for 3D models based on 3D Boolean operation. *Egyptian Informatics Journal*, 25:100436, Mar. 2024.

[14] Matthew Liberatore and William Wagner. Virtual, mixed, and augmented reality: a systematic review for immersive systems research. *Virtual Reality*, 25:1–27, Sept. 2021.

[15] Meshy LLC. Meshy - Free AI 3D Model Generator. `https://www.meshy.ai`.

[16] EU parliament Members. EU AI Act: first regulation on artificial intelligence, Aug. 2023.

[17] Harishankar Narayanan. codetiger/Font23D, July 2024. original-date: 2015-04-24T08:53:52Z.

[18] Stavros Nousias, Gerasimos Arvanitis, Aris S. Lalos, and K. Moustakas. Mesh Saliency Detection Using Convolutional Neural Networks. *IEEE International Conference on Multimedia and Expo*, 2020.

[19] T. O'Hailey. *Hybrid Animation: Integrating 2D and 3D Assets*. Focal Press, 2010.

[20] Pranav Parekh, Shireen Patel, Nivedita Patel, and Manan Shah. Systematic review and meta-analysis of augmented reality in medicine, retail, and games. *Visual Computing for Industry, Biomedicine, and Art*, 3(1):21, Sept. 2020.

[21] Fei Peng, Wenjie Qian, and Min Long. Visible Reversible Watermarking for 3D Models Based on Mesh Subdivision. In Xianfeng Zhao, Yun-Qing Shi, Alessandro Piva, and Hyoung Joong Kim, editors, *Digital Forensics and Watermarking*, Lecture Notes in Computer Science, pages 136–149, Cham, 2021. Springer International Publishing.

[22] Saeed Ranjbar Alvar, Mohammad Akbari, Lingyang Chu, Yong Zhang, et al. Amuse: Adaptive multi-segment encoding for dataset watermarking. *arXiv preprint arXiv:2403.05628*, 2024.

[23] Saeed Ranjbar Alvar, Mohammad Akbari, David Yue, and Yong Zhang. Nft-based data marketplace with digital watermarking. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4756–4767, 2023.

[24] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3D Deep Learning with PyTorch3D, July 2020. arXiv:2007.08501 [cs].

[25] Ahmad Rezaei, Mohammad Akbari, Saeed Ranjbar Alvar, Arezou Fatemi, and Yong Zhang. Lawa: Using latent space for in-generation image watermarking. *arXiv preprint arXiv:2408.05868*, 2024.

[26] Gursimran Singh, Tianxi Hu, Mohammad Akbari, Qiang Tang, and Yong Zhang. Towards secure and usable 3d assets: A novel framework for automatic visible watermarking, 2024.

[27] Francesca Uccheddu, Massimiliano Corsini, and Mauro Barni. Wavelet-based blind watermarking of 3D models. pages 143–154, Sept. 2004.

[28] Mythreye Venkatesan, Harini Mohan, Justin R. Ryan, Christian M. Schürch, Garry P. Nolan, David H. Frakes, and Ahmet F. Coskun. Virtual and augmented reality for biomedical applications. *Cell Reports. Medicine*, 2(7):100348, July 2021.

[29] Shengxian Wang, Li Li, Jianfeng Lu, and Ching-Chun Chang. A Watermarking Method for 3D Game Model Based on FCM Clustering and Density Tag Estimation of Vertex Set. In Lakhmi C. Jain, Sheng-Lung Peng, and Shiuh-Jeng Wang, editors, *Security with Intelligent Computing and Big-Data Services 2019*, pages 139–156, Cham, 2020. Springer International Publishing.

[30] Xiangyang Xu, Shengzhou Xu, Lianghai Jin, and Enmin Song. Characteristic analysis of Otsu threshold and its applications. *Pattern Recognition Letters*, 32:956–961, May 2011.

[31] Innfarn Yoo, Huiwen Chang, Xiyang Luo, Ondrej Stava, Ce Liu, Peyman Milanfar, and Feng Yang. Deep 3D-to-2D Watermarking: Embedding Messages in 3D Meshes

and Extracting Them from 2D Renderings, Mar. 2022. arXiv:2104.13450 [cs, eess].

[32] Innfarn Yoo, Huiwen Chang, Xiyang Luo, Ondrej Stava, Ce Liu, Peyman Milanfar, and Feng Yang. Deep 3d-to-2d watermarking: Embedding messages in 3d meshes and extracting them from 2d renderings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10031–10040, 2022.