TIM: A Sensitive-Parameter-Privacy Blind Watermarking Based on Zero-Knowledge Proof

Haoran Si Xi Lin Huiyan Chen
Beijing Electronic Science and Technology Institute

Abstract

Blind watermarking enables image ownership verification without requiring the original image. However, existing schemes usually require the owner to reveal the watermark seed and watermark positions during extraction. This creates a strong trust assumption on the verifier. Once such secrets are exposed, a malicious verifier can forge or remove the watermark. In this paper, we present TIM, the first publicly verifiable blind watermarking scheme with sensitive parameters privacy based on zero-knowledge proof. TIM reconstructs the extraction procedure of Integer DCT blind watermarking into an arithmetic-circuit-friendly form. This allows the owner to prove correct extraction without revealing the watermark seed or embedding positions.

TIM addresses three main challenges. The first is the high proving cost for high-resolution images. The second is the conflict between circuit rigidity and watermark robustness. The third is the hashing overhead of iterative state binding. To address them, TIM combines Nova and Spartan to decompose full-image extraction into iterative proofs. It uses a threshold-based voting mechanism for robust detection. It also introduces a hierarchical state update mechanism to reduce circuit and memory costs. Experiments show that TIM preserves good imperceptibility and robustness while achieving practical efficiency. For a 4K image, proof generation takes 5.61 minutes and uses 9.61 GB peak memory. These results show that TIM is the first blind watermarking scheme to achieve seed privacy, public verifiability, and practical performance for high-resolution image ownership proofs.

Keywords: Zero-knowledge proof, Incremental verifiable computation, Frequency-domain watermarking, Integer discrete cosine transform, Proof of ownership.

1 Introduction

With the rapid proliferation of social platforms and generative media, digital images are produced and redistributed at unprecedented scale, making provenance and ownership attribution an increasingly important problem. Blind watermarkings possess an inherent advantage in practical deployments because watermark extraction can be performed without access to the original image, and classical watermarking designs have studied this goal across spread-spectrum [1], QIM [2], DCT-domain [3, 4], SVD-based [5], and hybrid spatial-frequency

constructions [6]. However, existing blind watermarkings still require the image owner to disclose the watermark seed to the verifier during the extraction phase, where the seed determines the embedding information of the watermark values. This extraction mechanism implicitly relies on a critical security assumption: the verifier must be trusted. With the seed, an untrusted verifier can readily remove the watermark from the image, or forge watermarked images associated with this seed, thereby jeopardizing the copyright interests of the image owner.

In recent years, AI-based watermarkings [7] enable verification without requiring the disclosure of a seed, and have attracted considerable attention. However, their training datasets may be subject to poisoning attacks [8], and models may be vulnerable to adversarial manipulations, resulting in non-deterministic detection outcomes. Consequently, such approaches are unable to provide mathematically auditable security guarantees. Therefore, there remains a lack of a mathematically auditable watermarking scheme capable of effectively protecting the watermark seed and other sensitive information.

Zero-knowledge proof (ZKP) [9, 10] is a cryptographic primitive that enables a prover to convince a verifier of a statement without revealing the underlying secret witness, making it a natural foundation for privacy-preserving ownership verification. This property makes ZKPs inherently well suited for privacy-preserving image ownership verification scenarios, and non-interactive proof systems commonly rely on the Fiat–Shamir paradigm [11] to support public verification. Recent systems have explored ZKPs for attested images [12] and image-transformation verification [13, 14], showing that arithmetic-circuit proofs can provide auditable guarantees for visual media. A naive approach is that, takes the watermark seed and embedding locations as secret witnesses, with the watermark values derived from the seed. The watermarked image and extraction parameters are then provided as public inputs, and an arithmetic circuit is constructed to compile the entire watermark extraction procedure. Using the ZKPs, the image owner can subsequently convince the verifier of the validity of the extraction result without revealing the underlying secret witnesses. Moreover, due to the soundness of ZKPs, malicious prover that does not possess the correct seed cannot forge a valid proof.

It seems that ZKP simultaneously preserves the privacy of the watermark seed and enables publicly verifiable proof of ownership. However, this naive approach faces two fundamental challenges in practice.

The first challenge concerns computational performance. Modern high-resolution images, such as 4K (3840×2160) and 8K (7680×4320) images, contain an enormous number of pixels. Compiling the complete watermark extraction process for an entire image into a single arithmetic circuit would result in a constraint system whose scale far exceeds the practical capabilities of existing ZKPs. VerITAS [13] has demonstrated that proving an 8K image editing workflow using PLONK2 [15] requires nearly 130 GB of memory, such approaches impose substantial hardware requirements in practical applications.

The second challenge concerns robustness. Arithmetic circuits are inherently static and rigid, every constraint encoded in the circuit must be satisfied exactly. However, the robustness of blind watermarking requires that, the extraction algorithm be capable of recovering a sufficient watermark values even the watermarked image has undergone reasonable modifications, such as compression or minor editing, a requirement widely studied in robust spread-spectrum [1], QIM [2], and DCT-domain watermarking [3, 4]. Directly compile a

zero-error watermark extraction requirement into the circuit constraints implies that the entire proof becomes invalid even if only a single pixel of the image has been modified. There exists a fundamental tension between the rigidity of ZKPs and the robustness requirements of watermarking schemes.

Accordingly, we propose the central question addressed in this work:

Can we construct a blind watermarking scheme based on ZKP such that simultaneously preserves the privacy of the watermark seed and embedding locations, achieves a reasonable degree of robustness, and maintains practical efficiency?

To address the aforementioned question, we propose TIM (This Image Is Mine), the first blind watermarking scheme designed to reconcile privacy, robustness, and practical efficiency. In a high level, TIM adopts an Integer Discrete Cosine Transform (Integer DCT) watermarking, following the integer-transform design of H.264/AVC [16, 17] and prior integer-DCT watermarking systems [18]. TIM fully reconstructs its extraction algorithm as an arithmetic circuit, thereby enabling lossless watermark extraction over finite fields. To overcome the performance limit associated with a single large-scale circuit, TIM incorporates an Incremental Verifiable Computation (IVC) framework [19], decomposing the watermark extraction process for an entire image into a large number of fine-grained iterative steps. In each iteration, TIM processes a single row of 4×4 pixel blocks and incrementally accumulates proofs of computational correctness through the folding mechanism of Nova [20]. The Integer DCT inherently operates on a block-based structure, this decomposition strategy aligns naturally with the iterative paradigm of IVC. In the final stage, Spartan [21] compresses all intermediate folded proofs into a succinct non-interactive zero-knowledge (zk-SNARK) proof, following the broader line of PCP [22], IOP [23], and transparent proof-system techniques, enabling efficient public verification.

About robustness, TIM abandons the exact satisfaction of all constraints and instead introduces a voting mechanism. For each watermark-embedded image block, the circuit does not require the extracted result to be strictly identical to the expected watermark bit. Instead, the circuit contains a extract threshold. If the difference between the extracted coefficients of the current block falls within the prescribed threshold range, the block is counted as a *valid vote*; otherwise, it is counted as an *invalid vote*. The out-of-circuit verifier determines ownership based on the watermark error rate, which is defined as the ratio of invalid votes to the total number of watermarked blocks. While the ZKP guarantees only the correctness of the underlying computation and the reported voting results. The voting mechanism transforms *fuzzy tolerance* within arithmetic circuits, into a computable statistical decision process. This design follows the same high-level intuition as redundancy- and threshold-based watermark detection [1, 2, 6], where ownership is decided from aggregate evidence rather than a single exact coefficient comparison. It preserves the deterministic nature of circuit constraints while endowing TIM with robustness against signal-processing operations such as image compression.

About effciency, TIM requires that the input pixels in each iteration be binded into the hash state. For a 4K image, each iteration would involve more than 15000 hash operations, which is a substantial pressure for a circuit. To address this, TIM introduces a hierarchical state update mechanism based on random linear combinations. Under this mechanism, the pixels within each block are packed into a single field element through a cascaded encoding process, thereby reducing $16\times$ hash operations. Then, g blocks are aggregated into a single

digest value, and incorporated into the single hash, reduces $g\times$ hash operations. The security of this mechanism is guaranteed by the Schwartz-Zippel lemma [24].

In summary, the contributions of this work can be summarized as follows:

- We propose TIM, the first blind watermarking based on zero-knowledge proofs that provides seed-privacy. TIM fully compiles the extraction algorithm of an Integer DCT watermarking into an arithmetic circuit, enabling publicly verifiable image ownership proofs without revealing the watermark seed or embedding locations. It fundamentally eliminates the risks of watermark removal and forgery arising from seed disclosure.

- We systematically address the three challenges of privacy, robustness, and practicality. TIM incorporates the watermark seed and index as witnesses within the ZKP relation, leveraging the zero-knowledge to provide privacy. TIM provides redundancy for watermark extraction through voting mechanism, without compromising the determinism of the arithmetic circuit. With respect to practicality, TIM leverages IVC to decompose a large-scale circuit into a sequence of constant-size step circuits, and introduces the hierarchical state update mechanism, thereby enabling scalable proof generation for high-resolution images.

- We conduct extensive experiments on a range of standard resolutions images. The robustness evaluation shows that, under JPEG compression with a quality factor of $QF = 70$, TIM achieves a 480P watermark extraction error rate of 4.49%. The performance evaluation shows that, for a 4K image, TIM requires 5.61 minutes for proof generation, while verification takes only approximately 0.73 seconds. The experimental results demonstrate that TIM is the first blind watermarking scheme to achieve publicly verifiable watermark ownership verification with seed privacy protection at a practically efficient scale.

2 Related Work

Blind watermarking. Classical blind watermarking schemes including spread spectrum watermarking [1], Quantization Index Modulation (QIM) [2], and various DCT- and DWT-domain approaches [3–5]. All of them require the verifier to obtain the watermark seed during the extraction phase. Consequently, disclosure of the seed directly enables watermark removal or forgery, the verifier must be trusted.

ZKPs and multimedia verification. The application of zero-knowledge proofs (ZKPs) [9] to multimedia content verification has emerged as a promising research direction in recent years. VerITAS [13] applies PLONK2 to the verification of 8K image editing workflows. However, its monolithic large-scale circuit requires nearly 130 GB of memory, imposing substantial hardware demands and limiting its practical deployability. VIMz [14] further advances this research by IVC to realize arithmetic circuit representations of image editing operations.

Incremental verifiable computation. Incremental Verifiable Computation (IVC) [20, 25, 26] decomposes a long computation into a large number of uniform short steps,

which are subsequently compressed into a succinct proof through iterations. Nova [20] combines the R1CS instances generated at each step into a single relaxed R1CS instance, thereby avoiding the expensive pairing operations typically required in recursive SNARK constructions. Building upon Nova, Spartan [21] further provides zero-knowledge guarantees and proof succinctness. TIM does not require non-uniform IVC as employed by SuperNova [25], nor does it rely on the custom gates introduced in HyperNova [26].

Integer DCT. The floating-point coefficients used in the standard DCT prevent it from being represented exactly within finite-field arithmetic circuits. The Integer DCT introduced in H.264/AVC [16] replaces the floating-point DCT with an integer-kernel approximation, allowing all computations to be mapped lossless into addition and multiplication constraints over a finite field. Integer DCT watermarking schemes [6] have been extensively validated with respect to robustness; however, no prior work has integrated such schemes with ZKPs.

3 Preliminaries

This section summarizes the notation and background concepts used throughout the paper. We first introduce the notation for images, watermark values, and finite-field arithmetic. Then we review integer-based DCT watermarking [27], zk-SNARKs [9] and their circuit representation, and the incremental verifiable computation techniques [19] that underlie TIM's efficiency. These preliminaries establish the formal foundation needed to describe TIM's construction and security properties in later sections.

We first introduce commonly used notation. Let $\mathbb{F}_{\mathrm{p}}$ be a finite field, where p is prime. Let $p \in \{0, 255\}$ denote a pixel (luminance component), $\mathbf{p} \in p^{4\times 4}$ denote a (4×4)-sized pixel matrix (block), and $\vec{\mathbf{p}} \in \mathbf{p}^L$ denote an L-sized vector of blocks, where L is the number of matrices processed in a single IVC iteration. For an image $\mathbf{Img} \in p^{N\times M}$ (M and N are multiples of 4)[1], we denote

$$\mathbf{Img} = \begin{bmatrix} \vec{\mathbf{p}}_0 \\ \vec{\mathbf{p}}_1 \\ \vdots \\ \vec{\mathbf{p}}_{N/4} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_{0,0} & \mathbf{p}_{0,1} & \cdots & \mathbf{p}_{0,M/4} \\ \mathbf{p}_{1,0} & \mathbf{p}_{1,1} & \cdots & \mathbf{p}_{1,M/4} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{p}_{N/4,0} & \mathbf{p}_{N/4,1} & \cdots & \mathbf{p}_{N/4,M/4} \end{bmatrix},$$

and for a block $\mathbf{p}$, we write $\mathbf{p} = (p_0, p_1, \cdots, p_{15}) = (\vec{p}_0, \vec{p}_1, \vec{p}_2, \vec{p}_3)$. In this paper, we always set an iteration contains $L = M/4$ blocks and the iteration number is $T = N/4$.

3.1 Integer DCT Watermarking

TIM adopts the 4×4 integer discrete cosine transform (DCT) used by prior integer-DCT watermarking [18] and defined in the H.264 video coding standard [16, 17]. This transform maps a 4×4 pixel block to the frequency domain using a separable butterfly structure whose matrix consists entirely of small integer coefficients like $\{1, 2\}$. For a one-dimensional input

[1]The way resolution and matrix dimensions are expressed is opposite (e.g., resolution is given as width $\times$ height, whereas a matrix is expressed as rows $\times$ columns).

$\mathbf{x} = (x_0, x_1, x_2, x_3)$, the forward transform computes:

$$d_0 = x_1 - x_0,\ d_1 = x_3 - x_2,$$
$$L_0 = x_0 + \lceil d_0/2 \rceil + x_2 + \lceil d_1/2 \rceil, H_0 = d_0 + \lceil d_1/2 \rceil$$
$$L_1 = x_0 + \lceil d_0/2 \rceil - x_2 - \lceil d_1/2 \rceil, H_1 = d_1.$$

The 2D transform is obtained by applying this 1D butterfly row-wise, followed by the same butterfly column-wise on the intermediate result. The transform is integer-reversible, the inverse DCT recovers the original pixels exactly via the reverse lifting procedure, with no floating-point error.

In the frequency domain, each 4×4 block yields 16 DCT coefficients arranged from low frequency (top-left) to high frequency (bottom-right). To balance imperceptibility against robustness, the watermark is embedded in the mid-frequency region. Specifically, for each selected block, let mid-frequency coefficients $a = Y[1][2]$ and $b = Y[2][1]$ is retrieved, and the expected watermark bit can be inferred from the sign of $a - b$. When the watermarked image has undergone lossy processing such as JPEG compression, the coefficients are perturbed by quantization noise, and a decision threshold, is used to distinguish valid embedded signals from noise, following the standard robustness intuition of transform-domain watermarking [3, 4, 6].

A key challenge in realizing this transform inside an arithmetic circuit is that the division-by-two operations must respect integer truncation semantics rather than modular inversion over $\mathbb{F}_p$.

3.2 Zero-knowledge Succinct Non-interactive Argument of Knowledge

A zero-knowledge succinct non-interactive argument of knowledge (zk-SNARK) [9, 10] allows a prover to convince a verifier that it knows a witness $\vec{w}$ satisfying a public relation $\mathcal{R}(\vec{x}, \vec{w})$, without revealing any information about $\vec{w}$. The verifier checks a short proof π whose size and verification time are polylogarithmic in the computation size (succinctness).

Formally, a zk-SNARK satisfies: (1) completeness: an honest prover always produces a convincing proof, (2) zero-knowledge: the proof reveals nothing beyond the validity of the statement, (3) soundness: a malicious prover cannot forge a valid proof except with negligible probability, and (4) succinctness: the proof is short and verifiable in sub-linear time.

In TIM, the relation $\mathcal{R}$ encodes the entire watermark extraction procedure as an arithmetic-circuit relation, following the standard circuit representation used by practical ZK tool chains such as Circom [28, 29]. The public statement $\vec{x}$ consists of the watermarked image and the final accumulated error rate, the private witness $\vec{w}$ comprises the seed, the watermark index, and intermediate DCT shift witnesses. A successful proof thus demonstrates that the prover knows a seed under which the image yields a valid watermark, without exposing the seed itself.

3.3 Incrementally Verifiable Computation

Proving watermark extraction over a high-resolution image involves processing thousands of blocks—far beyond what a single flat circuit can accommodate. TIM addresses this through

incrementally verifiable computation (IVC).

Given an initial state z_0, a step function $\mathcal{F}$, and per-step witnesses $\vec{\mathbb{w}} = [\mathbb{w}_0, \mathbb{w}_1, \cdots, \mathbb{w}_{T-1}]$, IVC computes:

$$z_{j+1} \leftarrow \mathcal{F}(z_j, \mathbb{w}_j),\ j = 0, \cdots, T-2,$$

and produces, alongside the final state z_T, a proof Π_T that accumulates the correctness of all T steps. The verifier checks only Π_{final} with cost independent of T, which is property essential for scaling to large images.

TIM instantiates IVC using Nova [20], a recursive proof system based on folding. At each step, Nova folds the current running proof with the new step claim into a single relaxed R1CS instance of size $O(|\mathcal{F}|)$, where $\mathcal{F}$ is the number of constraints in the step circuit. This avoids expensive recursion overhead and does not require a trusted setup.

However, Nova does not guarantee zero-knowledge or produce a succinct final proof suitable for non-interactive verification. To obtain a full zk-SNARK, TIM wraps Nova's accumulated relation inside Spartan [21], a transparent proof system that adds the zero-knowledge and succinctness guarantees missing from the base iteration protocol. Spartan masks witness information through random linear combinations and enables the verifier to check only a small number of polynomial evaluations. The combined Nova+Spartan construction thus inherits Nova's efficient incremental proving while satisfying all four zk-SNARK properties of §3.2.

3.4 ZK-Friendly Hash Function

Standard hash functions such as SHA-256 [30] rely on bit-wise Boolean operations that translate to tens of thousands of constraints when arithmetized over a prime field. Poseidon [31] is a ZK-friendly hash function designed to operate natively over $\mathbb{F}_{\text{p}}$ via a sponge construction. A single Poseidon permutation consumes only a few hundred R1CS constraints, making it several orders of magnitude cheaper than SHA-256 inside a circuit.

TIM uses Poseidon for two distinct purposes: (1) it serves as the state transition hash within the IVC loop: each iteration updates the running state, cryptographically binding the entire block sequence into a compact accumulator, and (2) it acts as an in-circuit pseudorandom generator [32], ensuring that the expected watermark sequence cannot be predicted or biased without knowledge of s.

4 The Overview of TIM

This section provides a high-level description of TIM. It can be divided into three phases: (1) the watermark embedding phase, in which the image generator (e.g., a camera) obtains the original image and embeds a watermark, (2) the proof phase, in which the prover executes the arithmetic circuit implementing the watermark extraction algorithm and generates a proof about the extracted result, and (3) the verification phase, in which the verifier checks the validity of the proof to ensure the validity of the extraction result, and thereby ensure the validity of image ownership.

Figure 1 illustrates the overall protocol flow.

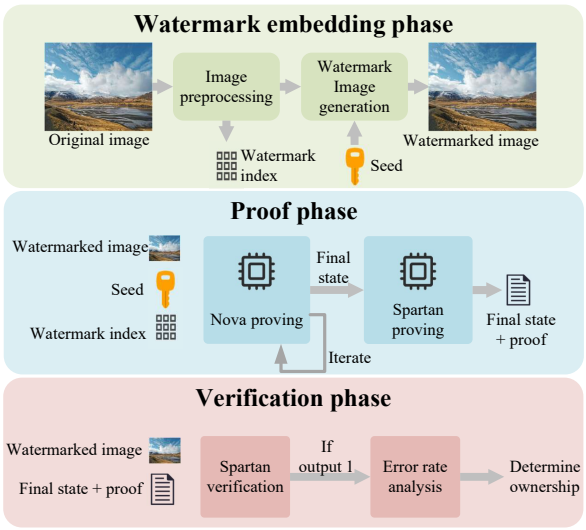

Figure 1: Overview of the TIM protocol.

4.1 Watermark Embedding Phase

The watermarked embedding phase includes image preprocessing and watermark image generation.

The image preprocessing aims to complete the preliminary work for image watermarking. Firstly, integer DCT watermarking requires only the luminance component Y in every pixel. TIM obtains the original image's luminance components by Reversible Color Transform (RCT [33]). Let every original pixel contains values R, G and B, computes

$$Y = \left\lfloor \frac{R + 2G + B}{4} \right\rfloor,$$
$$U_r = R - G,\ V_b = B - G,$$

where Y is the luminance, U_r is the red-difference chrominance and V_b is the blue-difference chrominance. The reason for choosing RCT is that it enables lossless color conversion, whereas the Irreversible Color Transform (ICT [34], $RGB \rightarrow YC_bC_r$) inevitably introduces errors during the conversion process. ICT is unacceptable for arithmetic circuit and ZKP, as any error in color conversion would lead to an incorrect state of the watermarked image, thereby causing verification failure. We transform each luminance Y as the corresponding pixel value p, and store U_r, V_b to restore the color of image.

Then, it selects blocks suitable for embedding the watermark value. Integer DCT watermarking requires avoiding embedding into regions with monotonic luminance distribution in order to guarantee imperceptibility, it adaptively traverses all blocks, selects those suitable for embedding, records them, and thereby obtains a watermark index set $\mathcal{I}_{wm}$, which contains L values, each being either 0 or 1. Every block in an iteration is assigned an index value, and 1 indicates that the block is watermarked.

During embedding, pixel values after inverse DCT may fall outside [0,255]. Conventional clamping is expensive to express in an arithmetic circuit. TIM therefore performs adaptive

pixel adjustment during preprocessing so that no overflow occurs for any watermarked block, eliminating the need for clamp logic inside the circuit entirely.

The watermark image generation is realized through integer DCT watermarking. It transforms **Img**'s blocks through §3.1 to obtain frequency domain values. We denote α is the embedding threshold what we embedded. We select two mid-frequency coefficients a and b of each block in $\mathcal{I}_{wm}$, adjust them such that $a - b = \alpha$ when the watermark value bit assigned to this block is 1, and $b - a = \alpha$ when watermark bit is 0. After embedding every blocks in $\mathcal{I}_{wm}$, frequency-domain blocks are first converted back to spatial-domain blocks via inverse DCT (§3.1), and then the image colors are restored by computing

$$G = Y - \left\lfloor \frac{U_r - V_b}{4} \right\rfloor,$$
$$R = U_r + G,\ B = V_b + G.$$

Finally, image generator outputs the watermarked image **Img**, and stores the watermark index $\mathcal{I}_{wm}$.

4.2 Proof Phase

Recall that disclosure of seed s would allow anyone to generate the watermark values, while disclosure of watermark index $\mathcal{I}_{wm}$ would reveal the watermark positions. In this phase, s and $\mathcal{I}_{wm}$ are taken as private witness. The proof therefore attests that the prover knows a seed and an index under which the watermarked image yields a valid extraction result, without ever exposing either secret to the verifier.

The arithmetic circuit is compiled from the watermark extraction algorithm. We decompose the extraction algorithm into an iterative function $\mathcal{F}$ adapted to Nova's IVC framework. We use IVC because the extraction workload is highly repetitive and naturally block-structured. Instead of proving one monolithic circuit for all image blocks, IVC keeps the base circuit tied to a single row of 4×4 blocks, reuses the same circuit across iterations, and folds the execution trace into a succinct final proof. This choice directly reduces peak circuit size and makes high-resolution proving feasible, while preserving the same public verification semantics through the final Spartan proof.

TIM maintains four state variables updated across iterations, denoted $z = (z_a, z_b, z_c, z_d)$, where

- z_a is a running Poseidon hash digest that binds every pixel to the computation, ensuring the prover commits to the entire image,
- z_b is a block counter, incremented once per traversed block,
- z_c is a watermark counter, incremented once per watermarked block (i.e., per block with $\mathcal{I}_{wm}[i] = 1$),
- z_d is an error counter, incremented once per watermarked block whose extracted coefficient relationship does not match the expected watermark bit.

In every iteration, $\mathcal{F}$ performs three tasks: (1) regenerate the watermark bit sequence from the seed s via hash function, (2) extract all blocks and evaluate the watermark decision rule on each watermarked block, and (3) update the four states for the next iteration. The detailed construction of $\mathcal{F}$ is provided in §5.

The watermark index $\mathcal{I}_{wm}$ produced during embedding (§4.1) serves as the block selection mask in the circuit. It only blocks with $\mathcal{I}_{wm}[i] = 1$ contribute to z_c and z_d. This index is provided as a private witness alongside the seed. A malicious prover could in principle supply an adversarially chosen index that discards failing blocks to artificially lower the error count. In the current design, we assume the prover is the legitimate image owner who has no incentive to weaken its own ownership claim, this honesty assumption is consistent like the C2PA-based trust model.

TIM extracts one row of blocks per iteration, producing a vertical sliding-window traversal. An overview of the iterative execution is illustrated in Figure 2. For an $M \times N$ image (both M and N multiples of 4), each iteration processes $L = M/4$ blocks, requiring $T = N/4$ iterations in total. This granularity balances the per-iteration extraction cost against Nova's folding overhead, and the row-aligned traversal naturally avoids partial blocks without additional padding logic.

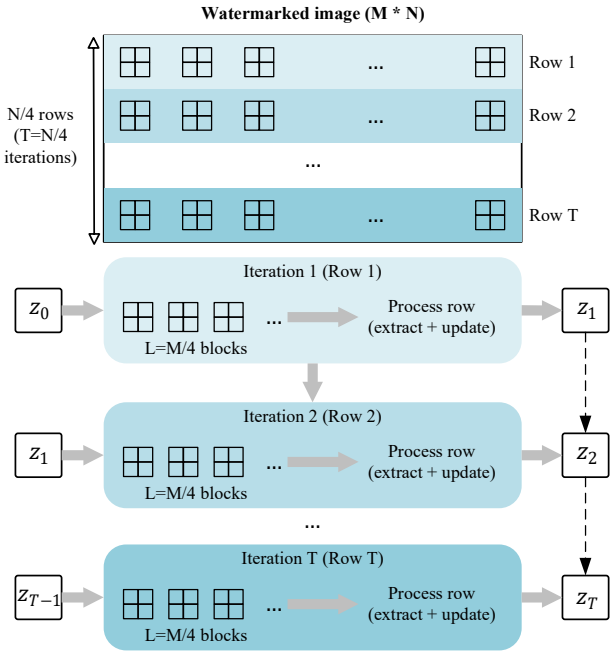

Figure 2: TIM's iteration.

After the final iteration, the circuit outputs the terminal state z_T, containing the complete hash chain and the three accumulated counters. The Spartan proof attests that z_T is the result of correctly applying $\mathcal{F}$ across all IVC iterations.

4.3 Verification Phase

In this phase, the verifier checks the Spartan proof π generated during the proof phase. A successful verification confirms that the prover honestly executed the function $\mathcal{F}$ in the arithmetic circuit and that the counters z_c and z_d faithfully reflect the extraction outcome. The verifier then computes the error rate z_d/z_c outside the circuit. If z_d/z_c falls below a pre-defined threshold (e.g., 30%), the watermarked image is considered valid, demonstrating that the prover holds knowledge of both the seed s and the watermark index $\mathcal{I}_{wm}$, and therefore holds ownership of the image.

5 TIM's Iteration Function

In this section, we present the Construction of TIM arithmetic circuit, which is compiled from the iteration function $\mathcal{F}$.

Recall that in TIM, the iteration function $\mathcal{F}$ should (1) generate the watermark value wm, (2) extract every block no matter it is watermarked, (3) determine whether the watermark is valid, increment the state z_d by one if it is invalid, and (4) update other states to ensure honest computation.

We first denote several functions as formal expressions to achieve the objectives described above: $\mathsf{WVG}_{\text{res}}$ is the watermark value generation function, its construction varies with the resolution of the image res, Ext is the watermark extraction function, used to obtain the difference of the mid-frequency coefficients in a block, H is the Poseidon hash function, and Cas is the cascade function.

The watermark value generate function WVG generates the same watermark value wm in each iteration instead of passing it. Although the Poseidon hash function H can be seen as a pseudo-random function to generate a watermark value, it always output in $\mathbb{F}_{\text{p}}$ and p is about 253-bit [35]. This means that when the width of the image M exceeds $253 \times 4 = 1012$, single hash H is insufficient to distribute watermark value bit to each matrix.

Therefore, for higher-resolution images, the generation of the watermark value relies on performing multiple H and assembling the hash values. We construct watermark generation function for each commonly used standard image resolutions respectively, i.e.,

$$wm \leftarrow \mathsf{WVG}_{\text{res}}(s, L), \tag{1}$$

where res $\in \{480\text{P}, 720\text{P}, 1\text{K}, 2\text{K}, 4\text{K}, 8\text{K}\}$, and L is the number of input blocks in one iteration, which is also the length of wm.

Generally, for a given image $\mathbf{I} \in p^{M \times N}$, the number of H operations required to generate its watermark value is

$$\text{num}_{\text{hash}} = \left\lceil \frac{M/4}{253} \right\rceil.$$

For instance, an IVC iteration of a 4K (3840×2160) image requires $\left\lceil \frac{3840/4}{253} \right\rceil = 4$ times H operations. Then, a 4K-size watermark value generation $\mathsf{WVG}_{4\text{K}}$ is shown as Algorithm 1. It generates a sufficient bits through 4 H operations (with state passing also occurring between the H), and distribute them in order to each pixel matrix.

Algorithm 1 The watermark value generation function WVG_{4K} for 4K standard image

Require: length $L = 960$, seed s, last iteration state z_{j-1},
Ensure: watermark value $wm = [\mathrm{bit}_0, \mathrm{bit}_1, \cdots, \mathrm{bit}_{L-1}]$

$\mathrm{num}_h = 4$ ▷ $4 \times 253 > \lceil 3840/4 \rceil$
$z = z_{j-1}$
for $i = 0$ **to** $\mathrm{num}_h - 1$ **do**
    $wm_i = \mathsf{H}(z, i)$
    $z = wm^i$
end for
for $i = 0$ **to** $L - 1$ **do**
    $\mathcal{I}_h = i \ / \ 253$
    $\mathcal{I}_b = i \ \% \ 253$
    $\mathrm{bit}_i = wm_{\mathcal{I}_h}[\mathcal{I}_b]$
end for
return $wm = [\mathrm{bit}_0, \mathrm{bit}_1, \cdots, \mathrm{bit}_{L-1}]$

For standard image, Table 1 shows the number of H operations (# Hash) required by WVG_{res} when each such image is used as input.

Table 1: The selection of circuit parameters for standard images

Std.	Resolution	L	#Hash	T
480P	640×480	160	1	120
720P	1280×720	320	2	180
1K	1920×1080	480	2	270
2K	2560×1440	640	3	360
4K	3840×2160	960	4	540
8K	7680×4320	1920	8	1080

The function Ext requires only the forward DCT to transform each 4×4 residual block from the spatial domain to the frequency domain. TIM adopts the integer DCT, whose transformation matrix consists purely of small integers and involves only additions and shifts, operations naturally compatible with arithmetic circuits.

Algorithm 2 formalizes the extraction procedure. For each block, a row-column separable butterfly computes the 4×4 DCT coefficient matrix Y. The two mid-frequency coefficients $a = Y[1][2]$ and $b = Y[2][1]$ are then selected as the watermark carrier pair.

Note that the integer DCT involves a right-shift operation (division by 2 with truncation toward zero). In a finite field $\mathbb{F}_\mathrm{p}$, the multiplicative inverse of 2 is $(\mathrm{p} + 1)/2$, which yields modular division rather than integer truncation. To faithfully replicate integer DCT semantics, the circuit introduces a shift index for each right-shift:

$$d = 2 \cdot q + r, \ r \in \{0, 1\},$$

where d is the difference to be shifted, q is the quotient (supplied by the prover and interpreted as $d \gg 1$), and r is the remainder constrained to be binary via $r(r-1) = 0$. This pair

of quadratic constraints exactly enforces integer division by 2 without relying on modular inversion, and the prover must supply the correct q and r to satisfy the constraints. The same technique is applied uniformly to all 16 shift operations (8 row-wise + 8 column-wise) in the DCT.

Algorithm 2 Watermark extraction function Ext

Require: Block $\mathbf{p} \in p^{4\times 4}$,
Ensure: The mid-frequency coefficients a and b

for $r \leftarrow 0$ **to** 3 **do** ▷ Butterfly for rows
  $x_0, \cdots, x_3 \leftarrow \mathbf{p}[r][0], \mathbf{p}[r][1], \mathbf{p}[r][2], \mathbf{p}[r][3]$
  $W[r][0] \leftarrow (x_0 + x_3) + (x_1 + x_2)$
  $W[r][1] \leftarrow (x_0 - x_3) + 2 \cdot (x_1 - x_2)$
  $W[r][2] \leftarrow (x_0 + x_3) - (x_1 + x_2)$
  $W[r][3] \leftarrow 2 \cdot (x_0 - x_3) - (x_1 - x_2)$
end for
for $c \leftarrow 0$ **to** 3 **do** ▷ Butterfly for columns
  $w_0, \cdots, w_3 \leftarrow W[0][c], W[1][c], W[2][c], W[3][c]$
  $Y[0][c] \leftarrow (w_0 + w_3) + (w_1 + w_2)$
  $Y[1][c] \leftarrow (w_0 - w_3) + 2 \cdot (w_1 - w_2)$
  $Y[2][c] \leftarrow (w_0 + w_3) - (w_1 + w_2)$
  $Y[3][c] \leftarrow 2 \cdot (w_0 - w_3) - (w_1 - w_2)$
end for
$a \leftarrow Y[1][2]$
$b \leftarrow Y[2][1]$
return a, b

After each Ext, TIM analyses a and b to determine whether this watermark is valid. Obviously, for a watermarked image that has not been tampered with or compressed, $a-b-\alpha = 0$. TIM allows the watermarked image to undergo limited editing while still enabling a sufficient number of valid watermarks to be extracted, i.e., robustness. Therefore, TIM adopts a relaxed decision rule. We define an *extraction threshold* β, with $\beta \leq \alpha$, to provide tolerance against editing, compression, and other benign distortions. The circuit then computes a vote value v according to the following decision rule:

$$v = \begin{cases} 1, & (2 \cdot \text{bit} - 1)(a - b) \geq \beta, \\ 0, & \text{otherwise}, \end{cases}$$

where $v = 1$ is valid, and $v = 0$ is invalid.

Elements of $\mathbb{F}_\text{p}$ represent residue classes modulo p and therefore have no intrinsic ordering. For example, the integer -1 is represented by the field element $\text{p} - 1$. As a result, the integer comparison in the decision rule cannot be evaluated directly in an arithmetic circuit over $\mathbb{F}_\text{p}$. We therefore convert the comparison into range-constrained algebraic relations that can be enforced inside the circuit.

First, assume that the DCT coefficients are bounded by a finite integer range, namely

$a, b \in [-o, o)$. We shift them into a non-negative representation:

$$\hat{a} = a + o, \quad \hat{b} = b + o,$$

where $\hat{a}, \hat{b} \in [0, 2o)$ can be range checked by $\mathsf{Num2Bits}(\lceil \log_2(2o) \rceil + 1)$. We then define the shifted coefficient difference as

$$\text{diff} = 2o + a - b.$$

Since $\text{diff} \in [0, 4o)$, it can also be range checked by $\mathsf{Num2Bits}(\lceil \log_2(4o) \rceil + 1)$.

Next, we choose a large power-of-two comparison boundary R. When bit $= 1$, the condition $a - b \geq \beta$ is equivalent to $\text{diff} \geq 2o + \beta$. When bit $= 0$, the condition $b - a \geq \beta$ is equivalent to $\text{diff} \leq 2o - \beta$. We encode the two cases as

$$X_1 = R + \text{diff} - (2o + \beta), \quad X_0 = R + (2o - \beta) - \text{diff}.$$

The branch selected by the watermark bit is folded into a single value:

$$X = \text{bit} \cdot X_1 + (1 - \text{bit}) \cdot X_0.$$

With R chosen large enough, $X \in [0, 2R)$. Let $\ell = \log_2(2R)$. We decompose X into ℓ bits:

$$X = \sum_{i=0}^{\ell-1} b_i 2^i.$$

The vote is set to the most significant bit, $v = b_{\ell-1}$. Because $R = 2^{\ell-1}$ and $X \in [0, 2R)$, $v = 1$ if and only if $X \geq R$. Thus, when bit $= 1$, $v = 1$ exactly captures $a - b \geq \beta$; when bit $= 0$, it exactly captures $b - a \geq \beta$.

The above construction realizes the original integer threshold comparison over $\mathbb{F}_\mathrm{p}$ through binary decomposition. The constants o and R are chosen to keep all intermediate values within the bounded integer range required for sound comparison in the field. In TIM, we set $o = 2^{11}$, which enables range checking via a 13-bit decomposition and prevents ambiguity from modular field representation. We set $R = 2^{16}$ so that the most significant bit of a 17-bit decomposition directly encodes the comparison result.

Each IVC iteration must bind every input pixel to the running state z_a so that the prover is committed to the entire image processed by the circuit. A straightforward per-pixel update,

$$z_{a,j} \leftarrow \mathsf{H}(p_{15} \| \mathsf{H}(p_{14} \| \cdots \mathsf{H}(p_1 \| \mathsf{H}(p_0 \| z_{a,j-1})) \cdots)),$$

would incur one Poseidon invocation per pixel. For a 4K image, this results in $4 \times 3840 = 15360$ hash operations per iteration, consuming approximately four million constraints and overwhelming the comparatively small cost of watermark extraction.

TIM therefore uses a two-level aggregation strategy. At the first level, each 4×4 block is packed into a single field element via Cas:

$$c = \sum_{i=0}^{15} p_i \cdot (2^8)^i.$$

Algorithm 3 The j-th IVC iteration function $\mathcal{F}^{(j)}$

Require: $\vec{\mathbf{p}} \in [\mathbf{p}_0, \cdots \mathbf{p}_{L-1}]$, extract threshold threshold, $(j-1)$-th state $z_{j-1} = (z_{a,j-1}, z_{b,j-1}, z_{c,j-1}, z_{d,j-1})$, seed s

Ensure: j-th iteration state z_j

$z_a, z_b, z_c, z_d = z_{a,j-1}, z_{b,j-1}, z_{c,j-1}, z_{d,j-1}$
$wm \leftarrow \mathsf{WVG}_{\text{res}}(s, L)$ ▷ Generate the watermark value
for $i = 0$ **to** $L - 1$ **do** ▷ Extract each pixel matrix
    $z_b = z_b + 1$ ▷ Update the state z_b
    $\text{bit}_i = wm[i]$
    $z_c = z_c + 1 \cdot \text{bit}_i$ ▷ Update the state z_c
    $a_i, b_i \leftarrow \mathsf{Ext}(\mathbf{p}_i)$
    $\text{diff} = a - b + 2o$
    $X_1 = R + \text{diff} - (2o + \beta)$
    $X_0 = R + (2o - \beta) - \text{diff}$
    $X = \text{bit} \cdot X_1 + (1 - \text{bit}) \cdot X_0$
    $v = \mathsf{MSB}_R(x)$ ▷ obtain the most significant bit
    $z_d = z_d + (1 - v)$ ▷ Update the state z_d
end for
for $i = 0$ **to** L/g **do**
    $r_c = \mathsf{H}(z_a, i)$ ▷ Fiat-Shamir challenge from the current state
    $\text{g} = 0$
    for $\text{k} = 0$ **to** $g - 1$ **do**
        $c_k = \mathsf{Cas}(\mathbf{p}'_{i \cdot g + k})$
        $\text{g} = \text{g} + c_k \cdot r_c^k$
    end for
    $z_a = \mathsf{H}(z_a, \text{g})$ ▷ Update the state z_a
end for
return $z_j = (z_a, z_b, z_c, z_d)$

This reduces hash invocations by a factor of 16. At the second level, every g consecutive packed values are aggregated through a random linear combination:

$$b = \sum_{i=0}^{g-1} c_i \cdot r_t^i, \tag{2}$$

where the challenge r_t is derived from the current computation transcript, e.g., $r_t = \mathsf{H}(z_a, t)$ for the t-th group. Here z_a is an intermediate value already produced by the IVC computation, and the group index t fixes the transcript position. Thus, r_t is deterministic inside the circuit but unpredictable before the corresponding transcript prefix is fixed, which follows the Fiat–Shamir heuristic for deriving non-interactive challenges from computation-dependent transcript values. The group digest b then updates the running state: $z_a \leftarrow \mathsf{H}(z_a, b)$. This further reduces hash invocations by a factor of g.

The binding guarantee of this aggregate scheme follows from the collision resistance of Poseidon and the Schwartz-Zippel lemma: two distinct sequences of packed values $c_0, \cdots, c_{g-1}$

and $c'_0, \cdots, c'_{g-1}$ produce the same b only if the transcript-derived challenge r_t happens to be a root of their difference polynomial, which occurs with probability at most $(g-1)/|\mathbb{F}|$, negligible in the field size.

The aggregation parameter g trades state-update granularity for constraint savings.

We give the complete version of the iteration function $\mathcal{F}$ that is shown in Algorithm 3.

6 Threat Model and Security Analysis

In TIM, the prover is the legitimate image owner, who holds the original image, the watermark seed s, and the watermark index index_{wm} produced during embedding. The verifier is any third party who wishes to verify ownership of a watermarked image $\mathbf{I}_{wm}$, but may behave maliciously by attempting to recover s or index_{wm} from the proof. An external adversary is a PPT entity who possesses the watermarked image but does not know s or index_{wm}, and whose goal is to forge an ownership proof that passes verification.

Both s and $\mathcal{I}_{wm}$ must remain hidden from the verifier and the adversary. Knowledge of s allows one to regenerate the full watermark bit sequence, and thus to forge watermarked images or to erase the watermark by inverting the embedding logic. Knowledge of the index reveals which blocks carry watermark information, enabling targeted removal or substitution attacks.

The adversary and the malicious verifier have access to the watermarked image, the proof π, and all public parameters of the proof system. We consider a single-proof setting: the adversary cannot collect multiple proofs from the same prover to perform cross-proof correlation analysis. The attack surface includes both cryptographic attacks seed recovery, index recovery, proof forgery, and signal-processing attacks aimed at watermark removal via compression, filtering, or transcoding.

We make the following assumptions. The prover is honest and performs embedding and proof generation faithfully, a dishonest prover gains no advantage from weakening its own ownership claim. The embedding phase itself is trusted, relying on the C2PA provenance chain to guarantee that the watermarked image indeed originates from the claimed prover [36]. On the cryptographic side, we assume Poseidon is collision-resistant in the random-oracle model [31], that Nova provides knowledge soundness for the incremental computation [20], and that Spartan provides zero-knowledge for the final proof [21].

TIM aims to satisfy the following properties:

- Completeness. An honest prover who holds the correct s and index_{wm} always produces a proof that the verifier accepts.
- Soundness. An adversary who does not know s or index_{wm} cannot produce a proof for $\mathbf{I}_{wm}$ whose accumulated error rate z_d/z_c falls below the verification threshold.
- Zero-knowledge. The proof reveals no information about s or index_{wm} beyond what can be inferred from the public output and the watermarked image.
- Succinctness. Proof size and verification time are independent of the number of image blocks.

The soundness reduces to the knowledge soundness of Nova and the collision resistance of Poseidon [20, 31]. Any valid proof must attest to a witness (s, index_{wm}) whose Poseidon expansion. An adversary who does not possess such a seed must either find a second preimage of the watermarked coefficient sequence under Poseidon, or forge a Nova proof without a satisfying witness. Both are infeasible under the stated assumptions. The zero-knowledge of the witness follows directly from the ZK property of Spartan, which ensures that the proof leaks no witness information [9, 21]. Completeness holds trivially when the prover supplies the correct inputs.

Multi-proof unlinkability is not claimed. In TIM, we only guarantee zero-knowledge for individual proofs. Attacks against the embedding phase itself (e.g., side-channel leakage of s during injection) are outside the scope of the proof system. Signal-processing attacks that degrade watermark detectability are evaluated experimentally rather than being addressed via cryptographic guarantees.

7 Experiments

This section reports three experiments: watermark image quality, JPEG robustness, and proof-system performance. We test whether TIM preserves visual fidelity, remains robust under compression, and keeps proving and verification costs practical on standard image resolutions.

7.1 Experimental Setup

We implemented TIM[2] by compiling the extraction procedure into an arithmetic circuit using Circom [28, 29], and then proving the row-wise iteration with Microsoft's Nova implementation [20, 37]. The final non-interactive proof is produced by Spartan [21]. All proof-related computations use the scalar field $\mathbb{F}_{\text{p}}$ of the Pasta elliptic curve [35], where $\text{p} \approx 2^{254}$. Poseidon [31] is configured with width 3, 8 full rounds, and 57 partial rounds, matching the in-circuit hash used both for watermark-bit generation and for state aggregation. All experiments were conducted on a machine equipped with an Intel Core i9-14900K processor and 48 GB of RAM, running Ubuntu 22.04.

Table 2: Standard images parameters

Std.	Resolution	L	T	Hash	g	WM blocks	α	β
480P	640×480	160	120	1	16	512	100	25
720P	1280×720	320	180	2	16	1024	105	25
1K	1920×1080	480	270	2	16	2048	110	25
2K	2560×1440	640	360	3	32	3072	115	25
4K	3840×2160	940	540	4	64	4096	120	25
8K	7680×4320	1920	1080	8	128	6144	125	25

[2]Our code available at https://anonymous.4open.science/r/TIM-399F/.

For the watermarking experiments, benchmark images are resized with bicubic interpolation to six standard resolutions: 480P (640×480), 720P (1280×720), 1K (1920×1080), 2K (2560×1440), 4K (3840×2160), and 8K (7680×4320). Table 2 reports the parameters used for each resolution. Here, L is the number of 4×4 blocks processed in one IVC iteration, and T is the number of row-wise iterations. The hash numbers follows the watermark value generation strategy in Table 1: wider images require more Poseidon outputs to provide one watermark bit for each processed block. The aggregation parameter g follows the state-update optimization in §5, increasing at higher resolutions to reduce the number of Poseidon state updates inside each iteration. The last three columns specify the number of embedded watermark blocks and the embedding/extraction thresholds used in the watermark decision rule. The qualitative comparisons in the image-quality and robustness discussion refer to standard transform-domain [3], quantization-based [2], spread-spectrum [1], and SVD-based [5] watermarking families.

7.2 Quality of Watermarked Image

We use PSNR [27] to measure watermarked-image fidelity:

$$\mathrm{PSNR}(I, I_{wv}) = 10 \log_{10} \left(\frac{255^2}{\mathrm{MSE}(I, I_{wv})} \right) \ \mathrm{dB},$$

where MSE is the mean squared error between the original and watermarked images. As shown in Table 2, with increasing image resolution, the ratio of watermarked blocks to the total number of blocks in TIM gradually decreases. Consequently, the alterations introduced by watermarking become increasingly imperceptible. For 480P, 720P, 1K, 2K, 4K, and 8K images, TIM obtains PSNR values of 38.78, 38.23, 42.47, 43.66, 46.02, and 46.90 dB, respectively. A watermark is considered visually imperceptible when its PSNR value exceeds 40 [27]. At lower resolutions such as 480p and 720p, this property is slightly compromised to ensure that a sufficient number of watermarks can be extracted. As the image resolution increases, images of 1K and beyond all achieve good imperceptibility. Figure 3 and Figure 4 show a 720P visual example.



Figure 3: Original image (720p).

About robustness, we selected 480p images with the highest watermark ratio, as it is more sensitive to compression. We evaluate JPEG compression on them using quality factors (QF) 100, 90, 80, 70, 60, and 50, and include non-watermarked fake-images to contrast.



Figure 4: Watermarked image (720p).

Table 3 reports the average error counter z_d and error rate. It show that over two-thirds of the watermarks are still successfully extracted when $QF = 50$. This means that TIM can still extract sufficient watermarks even the watermarked image is heavily compressed in a real-world scenario. Moreover, the fake image reaches 50.59%, close to random binomial distribution, can be clearly separated from valid images. TIM is not optimized purely for robustness; the trade-off is that its extraction remains integer-DCT based, branch-free, circuit-compatible, and seed-private.

Table 3: The results of robustness (480P)

Operation	Error counter z_d	Error rate (%)
$QF = 100$	0	0
$QF = 90$	2	0.39
$QF = 80$	2	0.39
$QF = 70$	23	4.49
$QF = 60$	111	21.68
$QF = 50$	146	28.52
Fake image	259	50.59

Table 4: Indicative comparison with representative watermarking schemes.

Scheme	PSNR (dB)	Robustness (%)	Privacy
DCT [3, 4]	42.00	95.00	No
QIM [2]	34.00	99.80	No
DFT magnitude [38]	66.40	99.96	No
DWT-DCT [39]	64.00	97.80	No
TIM (ours)	46.02	82.13	Yes

Then, we present a comparison of watermarked image quality between TIM and existing watermarking techniques [2–4, 38, 39] for 4K images. The result is shown in Table 4, where robustness is one minus the error rate. Compared with other watermarking techniques, TIM's robustness is not outstanding, this is because the arithmetic circuit cannot express

overly complex extraction operations, but the result is still acceptable. Moreover, TIM is the only scheme that does not require uploading the seed or the watermark index in Table 4.

7.3 Performance of Proving System

We next evaluate the cost of proving the extraction relation defined by the TIM circuit. Because TIM uses Nova, the image is not encoded as one monolithic circuit. Instead, each row of 4×4 blocks is processed by one iteration of $\mathcal{F}$, and the folded computation is finalized by Spartan. The evaluation therefore distinguishes prover complexity, which depends on the per-iteration circuit and the number of iterations, from verifier complexity, which benefits from the succinct final proof.

7.3.1 Prover's complexity

Table 5 and Figure 5 reports the circuit and proving costs for the standard resolutions. The circuit input consists of the public state $z = (z_a, z_b, z_c, z_d)$, the private seed s, the luminance pixels for the L blocks processed in the current iteration, and the quotient/remainder witnesses needed to express integer shifts and range checks inside the arithmetic circuit. Consequently, both the number of constraints and the input size are primarily controlled by L, while the total proving time also reflects the number of IVC iterations T.

Table 5: The experimental results of proving phase

Std.	Constraints	Inputs	Time (min)	Mem. (GB)
480P	19887	7845	0.71	0.57
720P	39774	15865	1.28	1.36
1K	59164	23525	2.71	2.85
2K	69391	31365	3.28	4.62
4K	96593	47045	5.61	9.61
8K	185941	94085	10.57	35.76

The results show a near-linear increase in circuit scale as the per-iteration width grows. The constraint count rises from 19,887 for 480P to 185,941 for 8K, while the input size grows from 7,845 to 94,085. This behavior matches the construction in Section 4: each additional block contributes a fixed amount of extraction logic and quotient/remainder witnesses, while the aggregation parameter g limits the growth of Poseidon-based state updates.

The proving time remains practical up to high resolutions. For 480P, TIM completes proving in 0.71 minutes with 0.57 GB peak memory. For 4K, the proving time is 5.61 minutes and peak memory is 9.61 GB, which is within the capacity of common workstation-class hardware. The 8K case requires 10.57 minutes and 35.76 GB peak memory. Compared with 4K, 8K roughly doubles the proving time but increases memory by about 3.7×, indicating that memory pressure becomes the main deployment constraint at the highest resolution. This gap is mainly explained by the geometry of the tested resolutions. An 8K image has twice the width and twice the height of a 4K image, so TIM processes twice as many blocks per iteration (L doubles) and twice as many row-wise iterations (T doubles). The doubled

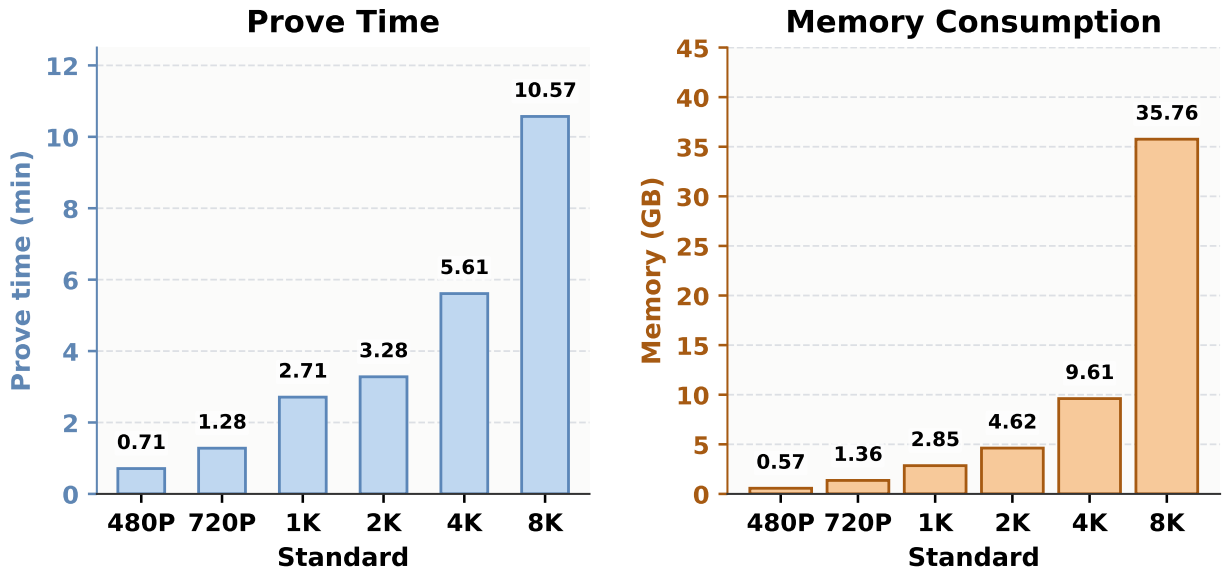

Figure 5: The experimental results of proving time and memory.

iteration count leads to an approximately doubled proving time, while the doubled per-iteration width substantially increases witness and folding memory, producing an empirical memory growth close to the expected $4\times$ total-pixel scaling. Thus, the 4K-to-8K results are consistent with TIM's near-linear dependence on image width per iteration and linear dependence on the number of iterations. For 8K images, peak memory can also be reduced approximately linearly by decreasing the number of blocks processed per iteration, at the cost of a proportional increase in proving time. This is consistent with TIM's design goal: proof generation is the expensive offline step performed by the image owner, whereas public verification remains lightweight.

7.3.2 Verifier's complexity

A key advantage of TIM is that verification complexity is independent of the image resolution and the number of IVC iterations. The verifier invokes Spartan's verification algorithm on the Spartan proof π to ensure the validity of extract result. By the succinctness of Spartan [21], the verification time is polylog($|\mathbb{C}|$), i.e., it grows only logarithmically with the circuit size. In our experiments, Spartan verification consumed approximately 0.75 s across all resolutions from 480P to 8K, confirming that the verification cost is essentially constant in practice.

8 Conclusion

This paper presents TIM, a privacy-preserving blind-watermark ownership verification scheme for images. TIM enables an image owner to prove that a watermarked image was correctly

generated and extracted without revealing the watermark seed, watermark index, or other private witness data. The main innovation of TIM is the integration of integer-DCT watermarking with zero-knowledge proofs. We reconstruct the watermark extraction process into an arithmetic-circuit-friendly form, use Poseidon to generate watermark bits and update states, and employ Nova with Spartan to support scalable proof generation over standard image resolutions. The experimental results show that TIM preserves image quality, remains robust under JPEG compression within the tested threshold, and keeps verification efficient. For 4K images, TIM achieves a PSNR of 46.02 dB, generates the proof in 5.61 minutes with 9.61 GB peak memory, and keeps verification at approximately 0.75 s. These results demonstrate that privacy-preserving public verification of blind-watermark ownership is practical for images.

Our future work will further enhance the robustness, and extend TIM beyond still images. In particular, TIM can be adapted to audio and video watermarking, enabling privacy-preserving ownership verification for broader multimedia content.

References

[1] I.J. Cox, J. Kilian, F.T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, 1997.

[2] B. Chen and G.W. Wornell. Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory*, 47(4):1423–1443, 2001.

[3] Mauro Barni, Franco Bartolini, Vito Cappellini, and Alessandro Piva. A dct-domain system for robust image watermarking. *Signal Processing*, 66(3):357–372, 1998.

[4] Chiou-Ting Hsu and Ja-Ling Wu. Hidden digital watermarks in images. *IEEE Transactions on Image Processing*, 8(1):58–68, 1999.

[5] Chin-Chen Chang, Piyu Tsai, and Chia-Chen Lin. Svd-based digital image watermarking scheme. *Pattern Recognition Letters*, 26(10):1577–1586, 2005.

[6] Frank Y. Shih and Scott Y.T. Wu. Combinational image watermarking in the spatial and frequency domains. *Pattern Recognition*, 36(4):969–975, 2003.

[7] Sven Gowal, Rudy Bunel, Florian Stimberg, David Stutz, Guillermo Ortiz-Jimenez, Christina Kouridi, Mel Vecerik, Jamie Hayes, Sylvestre-Alvise Rebuffi, Paul Bernard, Chris Gamble, Miklós Z. Horváth, Fabian Kaczmarczyck, Alex Kaskasoli, Aleksandar Petrov, Ilia Shumailov, Meghana Thotakuri, Olivia Wiles, Jessica Yung, Zahra Ahmed, Victor Martin, Simon Rosen, Christopher Savčak, Armin Senoner, Nidhi Vyas, and Pushmeet Kohli. Synthid-image: Image watermarking at internet scale. arXiv preprint arXiv:2510.09263, 2025. Accessed: 2026-06-11.

[8] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. arXiv preprint arXiv:1804.00308, 2021. Accessed: 2026-06-11.

[9] Justin Thaler. Proofs, arguments, and zero-knowledge. *Foundations and Trends in Privacy and Security*, 4(2-4):117–660, 12 2022.

[10] Maksym Petkus. Why and how zk-snark works. arXiv preprint arXiv:1906.07221, 2019. Accessed: 2026-06-11.

[11] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

[12] Daniel Kang, Tatsunori Hashimoto, Ion Stoica, and Yi Sun. Zk-img: Attested images via zero-knowledge proofs to fight disinformation. arXiv preprint arXiv:2211.04775, 2022. Accessed: 2026-06-11.

[13] Trisha Datta, Binyi Chen, and Dan Boneh. Veritas: Verifying image transformations at scale. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 4606–4623, 2025.

[14] Stefan Dziembowski, Shahriar Ebrahimi, and Parisa Hassanizadeh. VIMz: Private proofs of image manipulation using folding-based zkSNARKs. Cryptology ePrint Archive, Paper 2024/1063, 2024. Accessed: 2026-06-11.

[15] Polygon Zero Team. Plonky2: Fast recursive arguments with PLONK and FRI. `https://github.com/0xPolygonZero/plonky2`, 2022. Technical report in the Plonky2 GitHub repository. Accessed: 2026-06-11.

[16] H.S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky. Low-complexity transform and quantization in h.264/avc. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):598–603, 2003.

[17] G.J. Sullivan and T. Wiegand. Video compression - from concepts to the h.264/avc standard. *Proceedings of the IEEE*, 93(1):18–31, 2005.

[18] Amit M Joshi, Vivekanand Mishra, and R. M. Patrikar. Design of real-time video watermarking based on integer dct for h.264 encoder. *International Journal of Electronics*, 102(1):141–155, 2015.

[19] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *Theory of Cryptography*, pages 1–18, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[20] Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022*, pages 359–388, Cham, 2022. Springer Nature Switzerland.

[21] Srinath Setty. Spartan: Efficient and general-purpose zksnarks without trusted setup. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 704–737, Cham, 2020. Springer International Publishing.

[22] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of np. *J. ACM*, 45(1):70–122, January 1998.

[23] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam Smith, editors, *Theory of Cryptography*, pages 31–60, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[24] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *Symbolic and Algebraic Computation*, pages 216–226, Berlin, Heidelberg, 1979. Springer Berlin Heidelberg.

[25] Abhiram Kothapalli and Srinath Setty. SuperNova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive, Paper 2022/1758, 2022. Accessed: 2026-06-11.

[26] Abhiram Kothapalli and Srinath Setty. Hypernova: Recursive arguments for customizable constraint systems. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, pages 345–379, Cham, 2024. Springer Nature Switzerland.

[27] Bernd Jahne. *Image Representation*, pages 31–80. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[28] Marta Bellés-Muñoz, Miguel Isabel, Jose Luis Muñoz-Tapia, Albert Rubio, and Jordi Baylina. Circom: A circuit description language for building zero-knowledge applications. *IEEE Transactions on Dependable and Secure Computing*, 20(6):4733–4751, 2023.

[29] iden3. Circom: A zkSNARK circuit compiler. `https://github.com/iden3/circom`, 2022. GitHub repository. Accessed: 2026-06-11.

[30] United States. Superintendent of Documents. *National Institute of Standards and Technology*. US Government Printing Office, 1992.

[31] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. Poseidon: A new hash function for Zero-Knowledge proof systems. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 519–535. USENIX Association, August 2021.

[32] Elaine B Barker, John Michael Kelsey, et al. *Recommendation for random number generation using deterministic random bit generators (revised)*. US Department of Commerce, Technology Administration, National Institute of . . . , 2007.

[33] C. Christopoulos, A. Skodras, and T. Ebrahimi. The jpeg2000 still image coding system: an overview. *IEEE Transactions on Consumer Electronics*, 46(4):1103–1127, 2000.

[34] Int Std ISO, IEC JTC, et al. Information technology-jpeg 2000 image coding system-part 1: Core coding system. *ISO/IEC 15444-1*, 2001.

[35] Daira Hopwood. The pasta curves for halo 2 and beyond. *Retrieved June*, 17:2023, 2020.

[36] Coalition for Content Provenance and Authenticity. C2PA technical specification, version 2.2. Online specification, `https://spec.c2pa.org/`, 2025. Version 2.2. Accessed: 2026-06-11.

[37] Microsoft Research. Nova: High-speed recursive zero-knowledge arguments from folding schemes. `https://github.com/microsoft/Nova`, 2021. Rust implementation of Nova. GitHub repository. Accessed: 2026-06-11.

[38] Mohamed Hamidi, Mohamed El Haziti, Hocine Cherifi, and Driss Aboutajdine. A blind robust image watermarking approach exploiting the DFT magnitude. In *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, pages 1–6, 2015.

[39] Baisa L. Gunjal and Suresh N. Mali. Secured color image watermarking technique in DWT-DCT domain. arXiv preprint arXiv:1109.2325, 2011. Accessed: 2026-06-11.