Αλγοριθμικές Τεχνικές Υδατοσήμανσης Οπτικοακουστικού Υλικού

Κωνσταντίνος Μπαλής

Διπλωματική Εργασία

Επιβλέπων: Σταύρος Νικολόπουλος

Ιωάννινα, Ιούλιος, 2023

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA

Ευχαριστίες

Καταρχάς είμαι ευγνώμων στους γονείς μου και τον αδερφό μου για όλη την υποστήριξή τους σε όλη τη διάρκεια των μαθητικών και φοιτητικών μου χρόνων. Σε αυτό το σημείο, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Σταύρο Νικολόπουλο για όλη την βοήθεια και την καθοδήγησή του καθ’ όλη τη διάρκεια της διπλωματικής εργασίας. Ακόμη, θα ήθελα να ευχαριστήσω τους Ιωσήφ Πολενάκη και Βασίλη Βουρονίκο για την πολύτιμη βοήθειά τους. Τέλος θα ήθελα να ευχαριστήσω όλους τους φίλους και συμφοιτητές μου που έκαναν αυτή την πορεία ευχάριστη.

10/07/2023

Κωνσταντίνος Μπαλής

Περίληψη

Η ψηφιακή υδατοσήμανση βίντεο χρησιμοποιείται ευρέως για την προστασία πνευματικών δικαιωμάτων και τον έλεγχο ταυτότητας περιεχομένου. Αυτή η μελέτη προτείνει μια μέθοδο υδατοσήμανσης βίντεο που χωρίζει τα καρέ του βίντεο σε μη επικαλυπτόμενα μπλοκ και χρησιμοποιεί μια σειρά ακεραίων ως υδατογράφημα που κωδικοποιούνται ως 2D αναπαραστάσεις των αυτο-αντιστρεφόμενων μεταθέσεων τους, οι οποίες διαχωρίζουν περαιτέρω το μπλοκ σε κελιά. Η ενσωμάτωση του υδατογραφήματος λαμβάνει χώρα στον τομέα συχνότητας και τροποποιεί το μέγεθος των συντελεστών DC των αντίστοιχων κελιών. Ύστερα, χρησιμοποιείται μια συνάρτηση κατακερματισμού κρυπτογράφησης για την κρυπτογράφηση πληροφοριών σχετικά με τα καρέ που ενσωματώνονται σε συγκεκριμένα πλαίσια του ήχου. Στη συνέχεια, η μέθοδος ελέγχεται σχετικά με την πιστότητα και το ανεπαίσθητο του υδατογραφήματος στο βίντεο με υδατοσήμανση, την ικανότητά του να εξάγει τα υδατογράφημα από το βίντεο με υδατοσήμανση, την ικανότητα εξαγωγής των πληροφοριών του υδατογραφήματος από τα καρέ του σε περίπτωση επίθεσης σε ορισμένα από τα καρέ και τέλος την ικανότητά του να ανιχνεύει την ακεραιότητα των καρέ του βίντεο με βάση τις πληροφορίες από την τιμή κατακερματισμού.

Λέξεις Κλειδιά: Υδατοσήμανση Βίντεο, Διακριτός Μετασχηματισμός Fourier (DFT), Υδατοσήμανση Τομέα Συχνότητας, Αυτό-ανάστροφη Μετάθεση, Κατακερματισμός, Μπλοκ Ακεραιότητας, Πλαίσια Ήχου, Ακεραιότητα, Αυθεντικότητα, Πιστότητα, Αδιόρατο

Abstract

Digital video watermark is widely used to protect copyright and content authentication. This study proposes a video watermarking method that partitions the frames of the video into non-overlapping blocks and uses a series of integers as watermarks encoded as 2D representations of their self-inverting permutations which partitions further the block into cells. The embed of the watermark takes place in the frequency domain and modifies the magnitude of the DC coefficients of the corresponding cells. After that a cryptographic hash function is used to encrypt information about the frames that is embedded in specific frames of the audio. The method is then tested about the fidelity and imperceptibility of the watermark in the watermarked video, its ability to extract the watermarks from the watermarked video, the ability to extract the watermark information from its frames if some of them are attacked and lastly its ability to detect the integrity of the frames based on the information from the hash value.

Keywords: Video Watermarking, Discrete Fourier Transform (DFT), Frequency Domain Watermarking, Self-Inverting Permutation, Hash, Integrity Blocks, Audio Frames, Integrity, Authenticity, Fidelity, Imperceptibility

Table of Contents

1. Introduction

In this section will be discussed the general ideas of video watermarking and authenticity verification that are the main focuses of this study along with the contribution of this method and the previous works that influenced this this method will be presented.

1.1 Video Watermarking

Digital watermarking is the action of hiding a digital signature, a message, text, data, or a unique identifier (referred as watermark) in various media works like an image, audio or like in the case of this thesis videos. The primary objective is to safeguard intellectual property, to prevent copyright infringement, establish ownership, authenticate content, enable tracking and monitoring prevent.

A watermarking system is generally consisted of two parts. An embedding part responsible of taking the original media and embed the watermark. The watermarked media is then can be used instead of the original media, allowing the legal owner of the media at any given moment using a decoder to identify and extract the watermark and establish ownership of the media and so protect the copyright owner.

There are several criteria about how the watermarks can be classified, according to their purpose and applications. Two of the most important classifications are the classifications about the visibility and the domain the watermark is embedded. Watermarks can be visible or semi-visible, for example logos or text overlays in the frames of the video and therefore easily observable or invisible in which case the information is hidden within the video frames and is imperceptible to the human eye. Visible or semi-visible watermarks intentionally distort the original media in a way that is easy to identify ownership of the media. In contrast, invisible watermarks necessitate a decoding algorithm, during the extraction process, to locate and retrieve the hidden information from the video, thereby identify the legal owners.

Another way the watermarks can be classified are the domain the embedding process takes place. Spatial domain watermarking techniques embed the watermark into a video by directly modify the values of video frames or transform domain watermarking where mathematical structures are used to transform the spatial data to the transform domain or frequency domain. Transforms, in the case of digital watermarking are used to hide data in non-perceptible frequencies by modifying frequency coefficients.

Another classification of the watermarks involves the robustness, or the fragility of the watermark based on the level or desired resistance against attacks. For the purposes of this work, watermarking will be referred to imperceptible watermarking.

1.2 Authenticity Verification

Authentication is the process of attempting to verify the integrity and authenticity of watermarked videos to ensure their integrity and freedom from tampering. A digital watermark contains vital information that can be used to establish ownership of the media and validating that the video has not been altered in any way.

After the watermark has been embedded that uniquely represents the video's authenticity the owner should be able to reliably locate and extract the hidden information from the media at any given time, even if it has been subjected to some type of transformation for example, compression, scaling etc. which is often the case with videos and images on the Internet. To achieve this the owner of the video should use an appropriate algorithm, usually involving some kind of key, which was created during the embedding process, locate the watermarked areas, and extract the hidden information. The output of this process can prove the authenticity of the video if the watermark matches the expected output value or otherwise indicate potential tampering of the media. Authenticity verification is an extremely important process that provides reliable assessment about the integrity and authenticity of a video and its origin, thus facilitating identification and protection for the owners of the content.

1.3 Related Work

In this section, we will discuss the background of this work and present previous studies related to watermarking that have influenced the model in this work. Gwenaël Doërr provides a comprehensive overview around video watermarking in [1] discussing its applications, benefits, and challenges in digital watermarking, such as video processing, collision detection and the need for watermarking algorithms to have low complexity to meet real-time requirements. Also detailed analysis of trends in watermarking approaches are presented and compared. A theoretical review of various image watermarking techniques and their performances is presented at [2]. The review emphasizes the essential criteria that watermarking systems should fulfill, including robustness, imperceptibility, capacity, and security and furthermore, the assessment methods for evaluating watermarking systems are extensively covered. [3] and [4] offer detailed analyses of the objectives, goals, requirements, challenges, and significant applications of image watermarking. In [3], Mahbuba Begum and Mohammad Shorif Uddin, after presenting the requirements in designing watermark algorithms state that maintaining imperceptibility, robustness, and capacity simultaneously is an infeasible task and there is always a trade-off among those three requirements and the choice of a technique should be made by keeping balance among them. [4] provides a comprehensive summary of state-of-the-art watermarking algorithms, along with their limitations and potential solutions, offering an insightful overview of the watermarking research field.

Two works that were a main influence of this project were [5] and [6] proposed by Maria Chroni, Angelos Fylakis, Stavros D. Nikolopoulos and by Maria Chroni, Stavros D. Nikolopoulos, Iosif Polenakis, Vasileios Vouronikos respectively. In [5] the proposed method uses an integer as a watermark embedded into an image as the two-dimensional (2D) representation of a self-inverting permutation marking the corresponding blocks of an image in the transform dimension using Discrete Fourier Transform (DFT). In [6] the idea of [5] is further expanded by repeating the embedding process for the marked cells and offers much more security against various attacks.

Qingliang Liu et al. propose a blind, imperceptible, and robust digital video watermarking method [7]. Their approach optimizes the embedding position in key frames based on distortion in the wavelet transform domain. Additionally, [8] presents a blind image watermarking scheme that combines the strengths of both the spatial and frequency domains. It utilizes the unique properties of the direct current (DC) components obtained by the two-dimensional discrete Fourier transform (2D-DFT) in the spatial domain to embed and extract the watermark, thus enabling blind extraction.

Two image watermarking techniques that incorporate hash functions in their methods, which is critical in our work, are presented in [9] and [10]. In [9] a robust blind watermarking approach for medical images protection proposed by Fares Kahlessenane et al. applied in the frequency domain and uses a hash is used to encrypt patient information about the integrity of the extracted watermark in the least significant bit (LSB), with highly satisfying results in terms of imperceptibility and robustness. Muzamil Hussan et al. in [10] introduces a hash-based image watermarking technique for tamper detection and localization. The method divides the image into non-overlapping blocks uses Discrete Cosine Transform (DCT) to extract the DC coefficients and creates a cryptographic hash. Subsequently, bits of the hash are utilized as the watermark. Furthermore, Gunjan Nehru and Puja Dhar's paper [11] provides an extensive overview of audio steganography algorithms, including the crucial least significant bit (LSB) approach, which holds significance in our work. The paper describes LSB coding as a common method used for encoding a large amount of data. It also presents a robust and imperceptible audio data hiding algorithm.

1.4 Motivation and Contribution

The widespread distribution and accessibility of digital videos, ensuring the integrity and authenticity of video content has become a significant concern for content owners, distributors, and consumers. The unauthorized copying, tampering, download and uncontrolled distribution of videos for commercial profit present significant threats to intellectual property rights, copyright infringement, and the trustworthiness of digital media. To address these challenges, the development of effective methods for content protection and prevention of unauthorized usage is of utmost urgency. Video watermarking has garnered significant attention as an indispensable research domain for tackling these pressing issues.

Video watermarking research aims to develop advanced methods that embed imperceptible or semi-visible marks within video sequences, enabling authorized users to identify their content and detect any unauthorized modifications. This work presents an efficient watermarking technique for videos, building upon the works in [5] and [6], that utilize the 2D representation of self-inverting permutation for image watermarking. In this technique, a series of integers encoded as 2D representations of self-inverting permutation serve as watermarks for each frame of the video. During the extraction process, the owner can accurately locate the watermarked areas and employ the reverse algorithm used for encoding to precisely extract the series of integers embedded during the embedding process. Additionally, this work incorporates a hash-based function to embed hash value information about the frames of the video sequence as the least significant bits (LSB) of specific audio samples. This additional layer of protection indicates whether the video frames have been altered by comparing the hash value extracted from the images with the hidden hash value within the audio.

2. Theoretical Background

In this section will be discussed the main concepts that this method is built on. We will discuss the main components of the video that will be used in the method and the theories behind the encoding of the integers that will be used as watermarks, the watermarking in the frequency domain and the hash functions.

2.1 Video Components Analysis

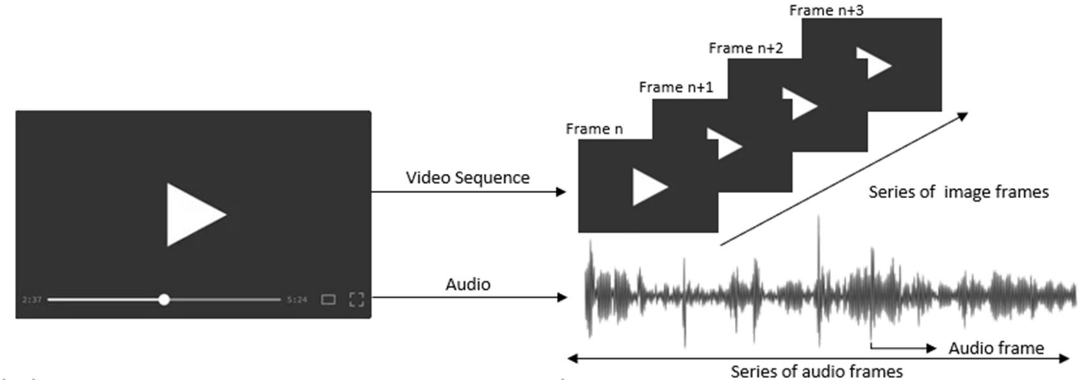
Videos are sequences of images used to represent a scene often combined by an audio. There are different elements and parts both visual and audio that make up a video and contribute to the overall experience. Understanding the different elements is essential to move forward with the processing of the video so in this section will be presented the necessary components of a video someone should know.

2.2.1 Audio-Visual Objects

Video sequence is the main component of a video that consists of a series of images, or how they are usually identified frames. When a significant amount of those frames are played in a brief period of time, they create the illusion of motion. Frames Per Second or simply FPS is a measurement that represents the number of frames that are displayed in a second to create such an illusion and is one of the most important characteristics of a video.

Audio is another key component of a video. Most videos have an audio that is basically data that are synchronized with the frames of the video to add to the overall experience. These data are separated into frames that contain information about the audio in certain moments. Audios and sounds in general can classified into two categories according to the number of channels used to reproduce the sound the mono and the stereo sounds. Mono (monoaural or monophonic) sounds are recorded using a single audio channel that can be heard identically from all directions. Stereo or stereophonic sounds on the other hand are using two separate channels meaning in can contain different information about the left and right channel providing a more realistic audio experience as it is capable to create the perception of direction and depth.

A video codec is an algorithm responsible for the way the data of a video are compressed and decompressed to reduce the file size of a video. That can involve either lossy or lossless compression and play a significant role to the quality of the video. Popular video codecs are H.264, H.265, AV1, MPEG-2 and MPEG-4. Similarly audio codes determine the quality of the audio data in a video how they are encoded or decoded and if the compression will be lossy or lossless. Popular audio codecs are MP3, FLAC, AAC and WAV. A container format or file format is the structure the video data have in a file and determines the way the data are stored, compressed, and organized. Different file formats offer different levels of compatibility and quality and support different codecs. Popular file formats involve MP4, AVI, MKV etc.



Audio-Visual Objects of Video

σ

2.2.2 Retrieving Image Frames

Retrieving image frames is the process of accessing specific frames from a video sequence for analysis and processing. More specifically in the case of this work this process is essential to get the frames that make up the video and add the watermark during the embedding phase, to create the watermarked video. Also, it is extremely import during the calculation of the hash so we can add information about the original video and therefore not require the original to verify its integrity. Similarly, during the extraction phase of the algorithm, we need to access those same frames by applying the reverse method embedding process to extract the watermark. That process usually involves loading the video sequence and extract the desired frame based on specific timestamps (Time-based extraction) that are calculated through the Frames Per Second (FPS) of the video.

2.2.3 Retrieving Audio Frames

Similarly retrieving audio frames is the process of isolating the audio data from the video to process of extract useful information from it. In this work we will use audio samples of the audio frames to hide information about the original video. Therefore, such a process is essential to manipulate the data of the audio separately from the video. A general process of extracting audio data from videos involves loading the video, isolating the raw data into a file, make any changes before composing the video sequence with the audio or simply to get information from its frames/samples.

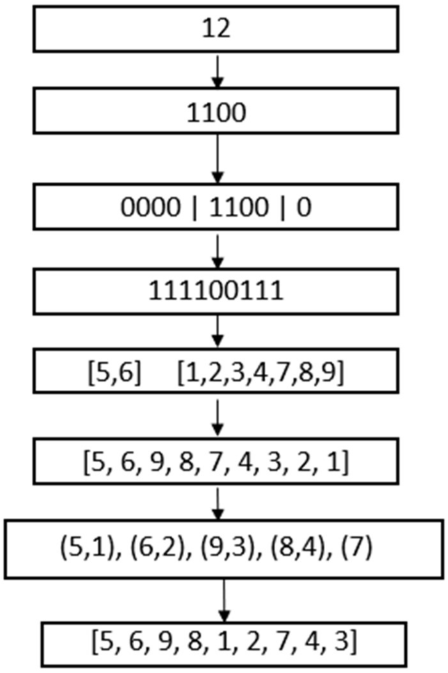
2.2 Integer Encoding as Self-Inverting Permutation (SIP)

Encode Integer as Self-Inverting Permutation

To watermark an integer into an image we first must convert it into a watermark capable of being embedded into a 2-dimensional (2D) object. For that purpose, we will use a 2D representation of self-inverting permutation. A permutation is simply a rearrangement of the elements of a set. For example, if we use a set like {1, 2, 3, .., n} for n = 9 a permutation could look like this: (5, 6, 9, 8, 1, 2, 7, 4, 3) where every element of the set appears exactly once.

By following this methodology presented below during the embedding process we can encode an integer as a self-inverting permutation meaning that following the reverse steps at the decoding process, we can get back the integer that was used as the watermark. Now will be presented the algorithm that will be used to encode integers into self-inverting permutations, with an example to make the process clearer.

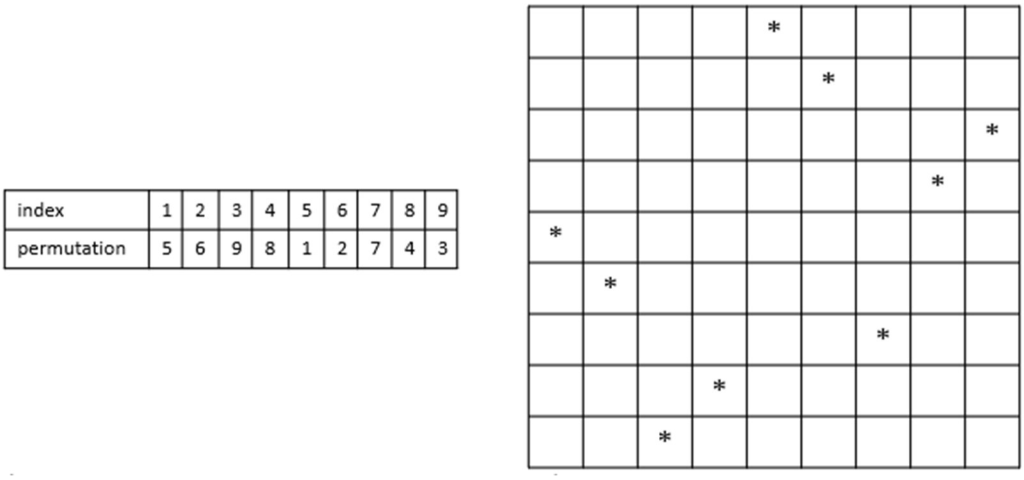
Given an integer for example eight, the binary representation of the integer is computed, in this instance 1000. After that, a binary sequence is formed by adding a sequence of zeros equal to the bits of the binary representation of the integer needs at the start of the binary representation and one more at the end of it. The binary representation now should look like this 0000|1000|0. Then the elements of the binary sequence are flipped, zeros become ones and ones become zeros (111101111). After that two sequences are created. The first one X contains the positions that have 0 in the binary representation, here only the fifth element is zero so X = [5] and the second Y has the rest of the elements, Y = [1, 2, 3, 4, 6, 7, 8, 9]. Then we construct the bitonic X|YR permutation where we have the elements of the first sequence in ascending order and the elements of the second one in descending order. The bitonic permutation of the example is [5, 9, 8, 7, 6, 4, 3, 2, 1]. Then cycles of length 2 are formed by pairing the first and last elements in the bitonic permutation (5, 1), the second and second to last (9, 2) etc. until we create those cycles for every element of the permutation. Note that if the number of elements in the permutation is odd, as it is in this example, the middle one will create a cycle of length 1. Lastly, we create the self-inverting permutation by adding the elements of the cycles into a list as follows: the first cycle is (5, 1) so the fifth element is 1 and the first one is 5, similarly about (9, 2) the ninth element is 2 and the second one is 9 etc. If there is a cycle of length 1 like in this example (6) this means that the sixth position will be 6. And like that the self-inverting permutation will look like this [5, 9, 8, 7, 1, 6, 4, 3, 2].



Encoding Process of Integer as Self-Inverting Permutation

2D Representation of the Self-Inverting Permutation

After encoding the integer, we should transform the permutation to create a representation capable of embedding the information into a 2-dimention item such as a frame. To achieve that we will create a grid of size $n \times n$ where n is the length of the permutation we created. In the previous example the grid the grid would be a 9×9. Then for each row i in the grid we will mark the cell in column that the ith position in the permutation indicates. For instance, the permutation we calculated in the previous example the corresponding cells would be for the first row the corresponding cell would be in the fifth column, for the second row it would be the ninth at so on. That grid represents the way we will watermark each. We will partition the block in $n \times n$ cells and for each of the corresponding cells that matches a marked position in the 2D representation of the SIP we will watermark those pixels with the method that will be presented in 3.1.5.



index	1	2	3	4	5	6	7	8	9
permutation	5	6	9	8	1	2	7	4	3

				*				
					*			
								*
							*	
*								
	*							
						*		
			*					
		*						

2D representation of Self-Inverting Permutation

Inverse algorithm

From 2D representation of Self-Inverting Permutation to 1D

During the extraction process we follow the inverse algorithm to get the integer back from the 2D representation of the SIP to prove the video is indeed ours by determining if the cells of the frames contain the embedded watermarks. After we successfully locate an area that we used for watermarking we will have the grid the represent the 2D permutation. By identifying the marked areas that contain the extra information we embedded and using positions regarding the row and the column that the mark is we should be able to reconstruct the self-inverting permutation.

Decode Self-Inverting Permutation to Integer

After that we must create the cycles, we described in the encoded process. Given a permutation, we will continue with the previous example to present the inverse method, [5, 9, 8, 7, 1, 6, 4, 3, 2] we iterate through the half of the list if number of elements in the are even else we iterate for half of the list and one more position. The cycles are formed by grouping the number in a position (i) with the number that is found in the ith position in the list. In this example the first number is 5 and is paired with the number in the fifth position 1. Therefore, the cycle is (5,1). After following this method for the rest of this list we get (5,1), (9,2), (8,3), (7,4) and 6. Then we form the bitonic permutation [5, 9, 8, 7, 6, 4, 3, 2, 1] and the form two sequences X and Y where X contains all the elements that are smaller than the next ones [5], as we observe only five is smaller than the next element and Y includes the rest [9, 8, 7, 6, 4, 3, 2, 1]. After that we create a list where add zeros in the positions of the list that are included in the X sequence and ones for the Y, [0, 0, 0, 0, 1, 0, 0, 0, 0]. We reverse the elements of the list [0, 0, 0, 0, 1, 0, 0, 0, 0] and at that point we have the binary representation of the integer we encoded in this case number 8.

2.3 Frequency Domain Watermarking

Frequency domain watermarking refers to the transformation of input data, such as an input image or frame, from the spatial domain (time) to the frequency domain using mathematical functions to embed the watermark. Discrete Fournier Transformation (DFT), Discrete Cosine Transformation (DCT), Discrete Wavelet Transformation (DWT) belong among the most common frequency transformation functions for frames. This section explores the key features of the frequency domain that make it more suitable for watermarking compared to spatial domain watermarking.

One of the key features of transformations is that they are reversable. By applying an inverse transformation, the data should return to their original values. One notable advantage of watermarking in the frequency domain is the distribution of watermark information across the entire marked cell upon inversing, making it more challenging for potential attackers to localize the marked cells and decode the embedded information, but easy for authorized users of the video who have information for the watermarking areas. Moreover, instead of changing pixel values directly which makes changes significantly more obvious the alterations are made in the transform coefficients. Those changes achieve the exact same goal but due to their nature of being made in the frequency domain they are barely noticeable to the human eye.

Both spatial and frequency domain watermarking have their advantages and their weaknesses. The spatial domain watermarking is simple but also vulnerable to attacks cause the watermark can but easier identified where frequency domain watermarking offers robustness and a better visual result which comes with higher computational cost. In this work that robustness and imperceptibility are the main focuses the choice will be the frequency domain watermarking.

2.4 Integrity Preservation (Hash)

The rapid growth of online data makes the employment of integrity preservation techniques necessary to safeguard against unauthorized modifications or tampering to protect content owners and their intellectual property. To ensure accuracy and consistency of a video over its entire life cycle we need to include such a function that guarantees that can detect even the slightest of changes that can be made to a video. For that purpose, we will include a cryptographic hash function in this work. Cryptographic hash function take data, such as frames in our case, as input and generate a fixed-size statistically unique output, known as a hash value, which is unique for a particular set of data and therefore are collision-resistant meaning that different inputs, even similar inputs with minor alterations, will always produce significantly different outputs.

The hash value serves as the digital fingerprint of the videos integrity and therefore this allows us to determine the integrity of a video by comparing the original hash value to the hash value of the received video. A match between the two hash values signifies that the integrity has been preserved while a mismatch raises concerns about its integrity and indicates potential changes or tampering of the videos data. Additional features of hash functions that make them ideal for integrity preservation of data are consistency which ensures that the function will always produce the same hash value for the same input data and generally being one-way functions, which means that there is no fast algorithm to restore the input information from the hash value offering an additional layer of protection against potential attacks to recover the input data.

3. The Model

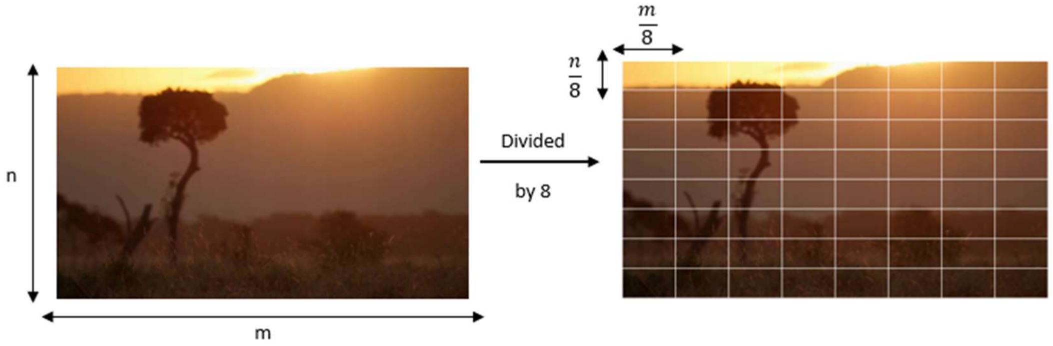
In this section the proposed algorithm will be presented. We will discuss how the watermark of the frames is accomplished but also the how that the embedding of the integrity information in the audio is achieved.

3.1 Preserving Video Authenticity

In this section is presented the algorithm and its techniques we use to embed the 2D representation of the self-inverting permutation that encodes an integer as watermark in the frequency domain of a frames and the decoding algorithm that locates the marked positions and extracts the watermark. Also, a hash-based method will be presented to include information about the watermarked video in the audio to ensure the integrity and allow the user to detect potential tampering.

3.1.1 Partitioning Image Frames into Integrity-Blocks

Analysis of video frames is a fundamental process in video watermarking. However, a video frames can be attacked and part of the information of the frame can be compromised and a watermark that depends on the integrity of the whole frame in most cases may prove to be unreliable. A solution to this problem involves the partition of the frame. Image or frame partitioning is the process of subdividing a digital image into multiple non-overlapping blocks along the rows and columns. This division allows us to treat each block independently to process specific regions within a frame. This allows us to ensure the integrity of blocks that have not been damaged even if other blocks within the frame have been. In the case of this work this process involves either the mark of the area or the extraction of the watermark. During the embedding phase of the method, we need to access specific areas to apply the 2D representations of the self-inverting permutation of the corresponding integer and mark specific cells while in the extraction phase, we need to access these same blocks to attempt to extract the watermark. The partition of the frame is done based on an input information of the user. The integer indicates the number of blocks that are created in each row and each column. An integer i partitions the frame into $i \times i$ blocks of size $\frac{n}{i} \times \frac{m}{i}$, where $n \times m$ is the size of the frame. A visual depiction of the partition can be seen in the following image.



Frame Partition into Integrity Blocks

3.1.2 Performing FDW in Image Frames

As we already discussed in 2.3 this method will embed the watermark in the frequency domain of each frame. The Discrete Fourier Transformation is an algorithm used to transform a signal from the spatial domain to the frequency domain. The Fournier transformation allows us to decompose a signal into a combination of sine and cosine waves that describe the frequency components.

The formula for the 2D Discrete Fourier Transformation of an N×M frame:

$$F(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x, y) e^{-j2\pi\left(\frac{ux}{N}+\frac{vy}{M}\right)}$$

where:

$F(u, v)$ represents the value at frequency coordinates (u, v)

$f(x, y)$ represents the pixel value at spatial coordinates (x, y)

N represents the number of rows of the frame

M represents the number of columns of the frame

j is the imaginary $\sqrt{-1}$

u in the range 0, 1,..,Nss-1 represents the horizontal frequency index

v in the range 0, 1,..,M-1 represents the vertical frequency index

Similarly given a $F(u, v)$ we can use the inverse Fournier Transformation to transform data from the frequency domain back to the spatial domain.

The formula for the 2D Inverse Discrete Fourier Transformation:

$$f(x, y) = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F(u, v) e^{j2\pi\left(\frac{ux}{N} + \frac{vy}{M}\right)}$$

After computing the Fourier transformation, we need to calculate the magnitude, which represents the amplitude of the corresponding frequency components of the transformation that will be used during the embedding and the extracting process.

3.1.3 Encoding Multiple Integers to Self-Inverting Permutations (SIPs)

By dividing the video into integrity blocks we need to establish the information that will be embedded. We have already touch on the way we will watermark the video, by calculating the self-inverting permutation that encodes an integer. In the case of this work, we will encode multiple integers to fill all the different blocks we have divided our frame. If the division of the frame creates for example n blocks, we need to encode n integers to watermark in each one of them. To achieve the encoding of the integers we will follow the methodology as describe in 2.2 to turn an integer into a representation capable of been embedded in a two-dimensional object such as a video frame.

The set of the integers we will encode will be random for each execution of the algorithm and proportional to the number of blocks allowing the user to include a unique and impossible to predict by a third-party watermark for every video. The set of integers are stored in a file for the user and serve as the key that verifies or not ownership of the video during the extraction process. After the integers are stored begins the permutation construction. Each integer is encoded to its self-inverting permutation following the steps of the algorithm in 2.2 and thus we have the necessary information to pass along to the algorithm responsible for embedding that information into each frame.

During the extraction phase we need to extract and get back the integers we originally embedded from the frames to verify the integrity of the video by following the inverse algorithm used for embedding also described in 2.2. By identifying the marked positions in each cell for every block in the frame we will get set of permutations. After applying to each of them the inverse algorithm we will get a set of integers that represent the extracted information from the watermark. An authorized user should be able to claim a video as his property by comparing the extracted set of integers with the integers that were stored during the embedding process.

3.1.4 Code-Based Multiple Watermarking

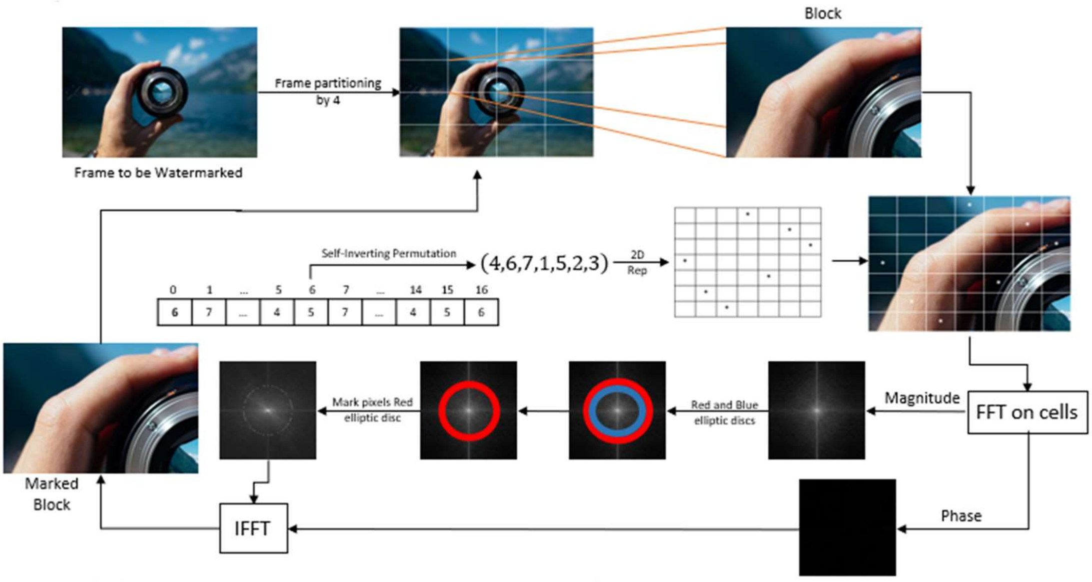
Multiple watermarking gives us the ability to embed multiple watermarks into a frame in certain areas and the ability to extract them individually or collectively. During the embedding process watermarks are inserted into different determined locations, self-inverting permutations of integers in different block in our case, by modifying the magnitude in the frequency domain, trying to minimize those changes to make the watermark imperceptible to the eye. The extraction process involves the location of the watermarked areas and the decoding of the embedded information.

The goal of the algorithm is to extract and the original watermark data and ensure the integrity preservation of a frame. An advantage multiple watermarking has is that the independency of the different marks allows us to extract even partial of the series of integers if part of the frame has been sustained damage in way that prevents the decoding algorithm from detect the watermark. Also, areas in the frame where the watermark has not successfully extracted can point to the area that have potentially been tempered or modified, where a single mark would completely fail.

In the proposed method multiple watermarking is achieved by dividing the frame into integrity blocks based on the input information the user gives about the number of blocks wanted in each row and each column. For example, the number 6 is given as an input the integrity blocks in each row and column will be six creating a total number of 36 blocks or generally $n \times n$ blocks where n is the block size. For the series of integers that will be utilized as watermarks in the integrity blocks based on the information about the number of blocks the algorithm will randomly generate $n \times n$ integers one for each block. To allow uniform embedment of watermarks in the frames the integers generated will be of a specific “class”, meaning the binary representations of the numbers in each execution of the algorithm will require the exact same number of bits. Remember that the 2D representations of the self-inverting permutation of the integers creates a grid of size $m \times m$ where m is equal to 2 * (integer binary size) + 1. Therefore, during the execution of the algorithm the user enters as a parameter the desired number of bits the binary representations of the integers used in each case will have.

3.1.5 Embedding Codes to Frames

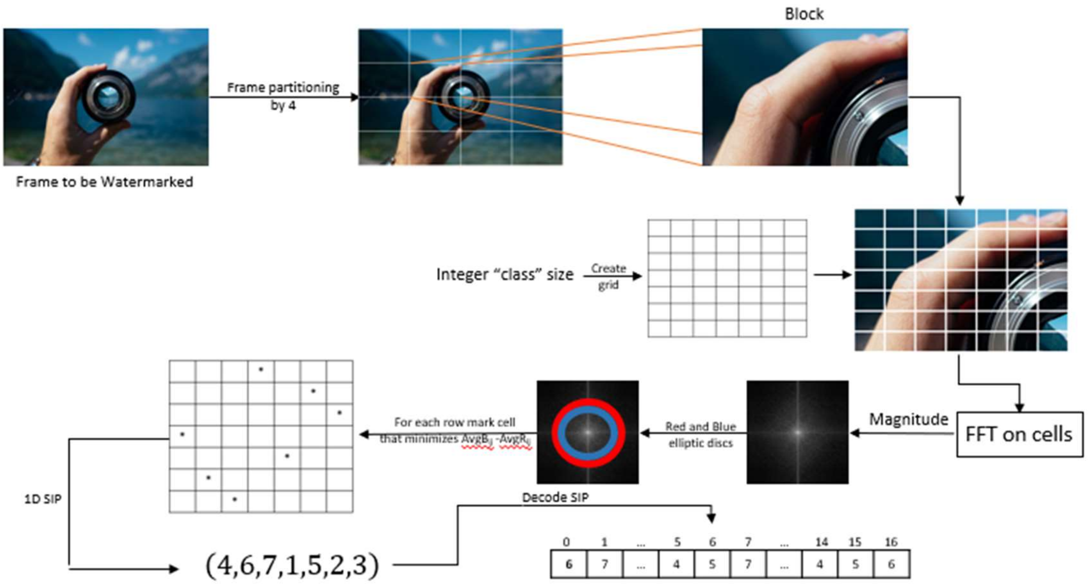
The next step in this method is the marking of the areas. After the partition of the frame into integrity blocks and multiple encodings of integers as 2D representations of their self-inverting permutations we can accurately find the corresponding cells of each block we will watermark. Those cells are spitted into their RGB channels. For each channel separately we compute the DFT magnitude of the cells, as we already discussed the marking of the cells is inserted in the frequency domain of the image, and to mark the area we will utilize two ellipsoidal Annuli, one outer "Red" and an inner "Blue" with the following properties. Both are concentric at the center of the magnitude with widths Pr and Pb, with radiuses for the "Red" R1 and R2 in y and x axis respectively while "Blue" has (R1 - Pr) in y and (R2 - Pb) in x which means that the two ellipsoidal annuluses touch each other. By applying the two ellipsoidal Annuli we create two "groups" of pixels the ones that fall in the "Red" area and the ones in the "Blue" area. We compute the average values of the pixels in each area of each cell of the block, AvgRij and AvgBij. After that compute, the value Dij for each cell. Dij is equal to |AvgBij-AvgRij| if AvgBij ≤ AvgRij, or 0 otherwise. After computing each Dij in a row keep the maximum MaxDi of the row. The marking of the corresponding cell of the 2D representation of the self-inverting permutation in each row is happening by increasing the magnitude matrix by AvgBij – AvgRij + MaxDi + c, where c is a value, the user enters when the algorithm is executed. The last step is to apply the inverse DFT to the cell to get back the image in the spatial domain.



Embedding Process of Codes to Frames

During the extraction process a similar approach will be used based on the “Red” and “Blue” ellipsoidal annuli to detect the areas that were marked in the embedding phase of the algorithm and extract the 2D representation of the self-inverting permutation in each block. After the frames have been partitioned once again into integrity blocks and based upon the information the user has entered, about the bits of the class of integers used in the embedding process, during the execution of the algorithm a grid is constructed. For each row in the grid a cell will be marked by the extraction process and if the correct cells are marked this grid will be the 2D representation of the integer used in the embedding process in each block of the frame.

For each cell we compute the DFT transformation and get the magnitude. For each magnitude matrix place the ellipsoidal annuli as previously described and get the average values in the two areas, AvgRij and AvgBij for the “Red” and “Blue” respectively. In the grid mark the cell in each row that the AvgBij – AvgRij is minimum. After that following the inverse algorithm in 2.2, we can compute the self-inverting permutation and then the integer for every block and get back the series of integers.



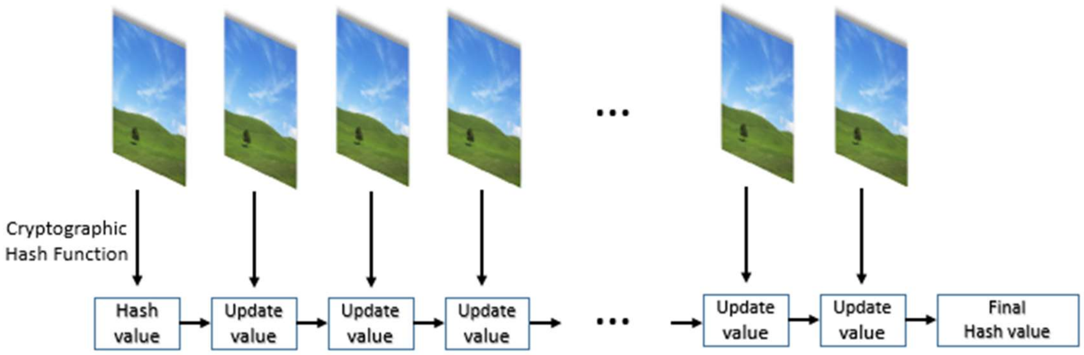
Extraction Process of Codes from Frames

3.2 Preserving Video Integrity

Integrity preservation is crucial technique to ensure authenticity for videos. In the context of integrity preservation in our work a hash function will be utilized, which is extremely effective to detect modifications in data as it produces a statistically unique output or hash value for the data, serving as the digital fingerprint, of our video and has many other desirable properties as already discussed in 2.4. Furthermore, we will present the algorithm that will be used to embed this information into our final video and how it will be used to determine the integrity of the video.

3.2.1 Hash-Based Video Frame-Chain

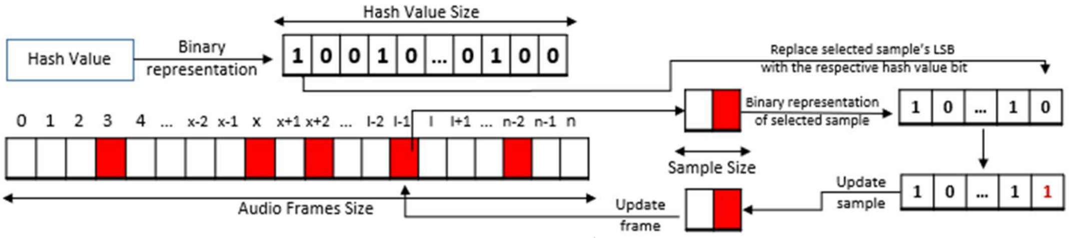
To ensure the integrity of the video we need to gather information for all the frames in encrypted form and embed that information within the watermarked video, allowing the user to compare at any point of the life cycle of the video, the hash of the frames with that hidden information of the original video to verify its integrity. For that purpose, we will follow a block-chain approach which links the frames together in chronological order to form a chain. In the hash-based technique that means each time we update the hash value so we can have a reference to the hash of all the previous blocks up to that point. The resulting hash value uniquely represents the information about the frames of the original watermarked video providing transparency and security to the owner of the video.



3.2.1 Embedding Integrity Information into the Least Significant Bit (LSB) of Audio Samples

After the computation of the hash value of the frames follows the embedding of the information into the video. The choice in this work is the audio of the video and more specifically the least significant bits (LSBs) of samples in certain frames of the audio that will be chosen randomly the number of which is determined by the size of the hash used. In our case the hash value will be 64 bytes long (256 bits) therefore there will be chosen 256 distinct frames that the information will be embedded. Randomly will be chosen the samples of the audio too and the information about the frames and the channels that were chosen are stored so the owner of the video can have the necessary information to locate the positions and extract the embedded information. It is obvious that if the audio if mono meaning one sample per frame for both the left and the right audio channels the information will be embedded into that single sample and there will be no need to store information about the choice of sample.

First, we need to transform the hash value for its hex form into bits in order to embed it in the least significant bits of an audio frame. Then for each selected frame and its selected sample we get the binary representation of the sample, replace the last bit (LSB) with one of the bits of the binary representation of the hash and then convert the value back to its original representation form and update the audio sample. By following the inverse algorithm, we can trace the samples that contain the bits that represent the hash value with the stored information about the marked samples and frames, extract the bits and use the hash value to verify the integrity of the video.



Hash Value Size
Hash Value
Binary representation
1 0 0 1 0 ... 0 1 0 0
Replace selected sample's LSB with the respective hash value bit
0 1 2 3 4 ... x-2 x-1 x x+1 x+2 ... l-2 l-1 l l+1 ... n-2 n-1 n
Binary representation of selected sample
1 0 ... 1 0
Sample Size
Update sample
1 0 ... 1 1
Audio Frames Size
Update frame

The Embedding Process of Integrity Information in the LSB of Audio Samples

4. System Architecture

In this section will be presented the architecture overview and we will discuss the key components of the of the proposed algorithm.

4.1 System Components

Now that the main concepts of the model have been analyzed, in this section will be described the architecture of this work. The main components of this work are:

Video Object Management

To begin the embedding process of the algorithm firstly we need to load the video by utilizing a library that allows video editing and video processing tasks such as (MoviePy or OpenCV). Using an appropriate command, we can create a video clip originating from the file the user gave during the execution of the algorithm. After creating the video clip, we can access its components. For the visual components of the video a necessary attribute is the videos' frames per second (FPS), which gives us information about how many separate images we need to extract in every second of the video clip. We can also access the audio clips of the video and getting the parameters for the audio like the frame rate and the number of channels we can access the audio frame-by-frame and extract them.

After the processing of the frames is finished, we need to form the video sequence back from the individual frames using the information about the FPS of the videos. Lastly, after forming the video sequence and complete the processing and the audio we gather the two components to reconstruct the output video using the appropriate codecs for video and audio according to the desired quality of the output video. For example, in our case that we embed sensitive information in the audio of the video by marking the least significant bit of audio frames we must use lossless audio codecs and by extension a video codec that supports it, something that has direct impact on the size of the audio file and its format.

Partitioning Visual Components to Blocks

As already mentioned, to embed the watermark into the frames we will partition the frame into non-overlapping integrity blocks. Therefore, the next key component of the algorithm is to divide the frame using the input information the user gave for the division of the frame. That creates a $n \times n$ grid on the frame if n is the input integer. From that point on we proceed with the embedding of the codes and after finishing that we merge the individual cells to form back the image that now contains the watermark information.

Partitioning Blocks into Cells and Embed Codes

The next step after partitioning the image into integrity blocks is to embed the watermark. From each of the channels (RGB) of the image we create the 2D representation of the self-inverting permutation that corresponds to the cell. We apply the 2D grid in the block and for each the marked cells in the grid we execute the algorithm presented at 3.1.5. The last step in this component is to reconstruct the block using the individual cells and get the watermarked the block. After finishing the processing of all the blocks, we can successfully reconstruct the watermarked frame.

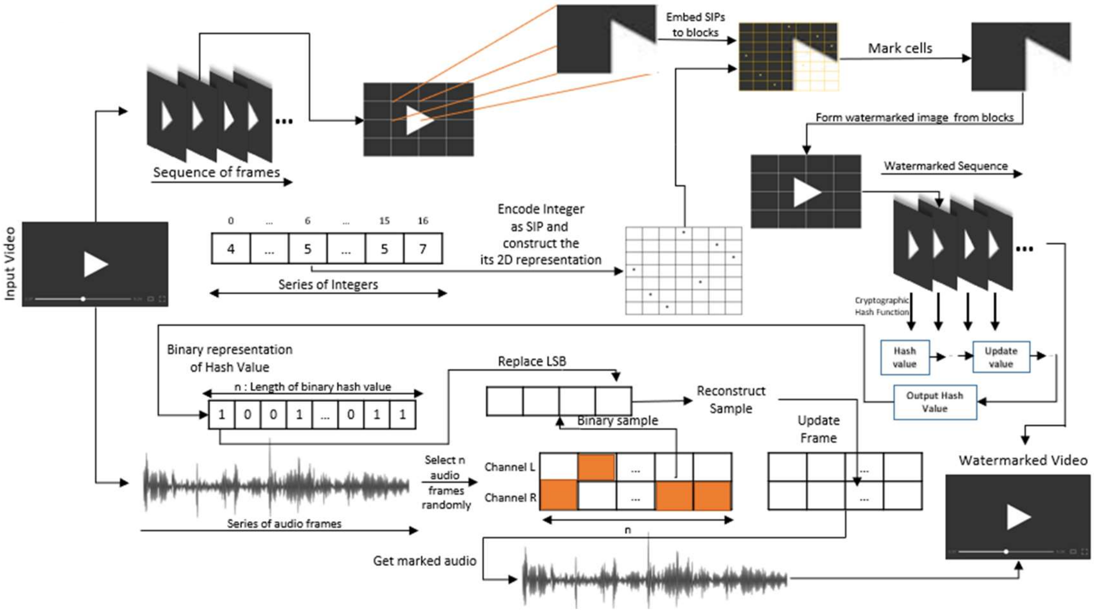
Integrity Chain

Integrity chain is the process we follow to link the frames of the video following the block chain approach after the watermarking process. Frame-by-frame we give each of the frames as parameters in a cryptographic hash function (SHA-256 in this case) to create a statistically unique hash value for that particular set of frames.

Embed Frame Chain into Audio Frames

After successfully reading the audio data and calculate the hash value we are ready to execute the last main components of the method. That involves marking specific sample of audio frames. We select randomly as many frames and samples in each one, as the bits of the binary representation Then the process of embedding is described in 3.2.2 by replacing the least significant bit of each selected sample in the frame.

4.2 Integration and Deployment



Architecture Overview

In this section will be presented a short overview of the watermarking technique, presenting the whole flow of the algorithm from the moment we take the input video up to the point where we have the watermarked video summarizing everything that was presented about the model.

When the algorithm is executed begins the management of the components of the video, where the sequence of frames and the audio are extracted. The next step is to retrieve each of the frames of the video from the sequence and partition them into integrity blocks based the input information of the user. Based on that information a series of integers will be randomly created consisted of integers of the same "class" meaning that their binary representations need the same number of bits to be represented. Each of the integers are encoded into the 2D representations of their self-inverting permutations based on the encoding method that was presented in 2.2. Then we apply the grid of each integer that is the 2D representations of the SIP to the corresponding block and following the method presented in 3.1.5 we can watermark the cell in the transform domain using the value the user entered. After doing this process for every frame of the video, a watermarked video sequence can be created.

Except the extraction of the watermark, this work focuses on preserving the integrity of the frames. For that reason, we need to keep information about the frames and to accomplice that a cryptographic hash function is utilized. Every single of the frames a given as parameters sequentially in the hash function to produce a statistically unique value that is updated thus keeping information about its previous frames, an integrity chain. We form the binary representation of the hash value and randomly select frames equal to the number of bits in the binary representation of the hash and the same equal number of samples to embed the frame chain. The least significant bits of these samples are replaced, and the frames of the audio marked. Lastly, we combine the video sequence with the marked audio to form the watermarked video.

The extraction phase of the method has must in common with the embedding. The video object management is obviously once again the first main component of the process to get the frames and the audio from the video. Using the information, the user gave during the execution of the algorithm the are partitioned into blocks. We put a grid of size that is indicated by the information stored about the keys and the extraction process described in 3.1.5 is used to decide the marked positions of the grid. If these positions are successfully found, we can decode the integer that was used and match the stored information we have from the embedding process.

As for the integrity preservation, parallel to the extraction of the SIPs the frames are given as parameters in the same cryptographic hash function and produce the hash value of the video frame chain. Using the information stored in the embedding process about the marked frames and samples in the audio, these positions are located, and their least significant bits are extracted. Now the hash value can be reconstructed from its binary form and compared to the hash value of the frames. If the two hashes match, we can prove that integrity has been preserved.

5. Evaluation

In this section will be discuss how we will evaluate the proposed method (the metrics and the attacks), the experimental results will be presented, and we will also compare this method to similar studies.

5.1 Dataset

To test the watermark scheme of this study we obtained two videos obtained from YouTube, specifically selecting videos labeled as free for use or without copyright restrictions by their creators. The videos are in mp4 file format with different resolutions. The first video "Nature" is of 1080p resolution (1980×1080), 5 seconds long and depicts different landscapes and the second one is also a 5 second clip of a man staring at the sea and this video is of 720p resolution (1280×720).

5.2 Attack Vector

To test the robustness of the proposed method a variety of attacks were utilized. An attack vector is a path or means by which an attacker can maliciously harm, in this case, a video. These attacks can target the integrity, availability, and confidentiality of the video content. Below we describe the different attacks that we applied in our dataset.

(1) Noise Attacks. Noise attacks are a type of video frame manipulation technique where random and unwanted variations, referred to as "noise" are added to the original content. Two common types of noise attacks that were used in the case of this evaluation are Gaussian noise and Salt & Pepper noise.

Gaussian noise is a type of statistical noise that follows a Gaussian or normal distribution. It appears as random variations in brightness or color values in the frames of the video independently. The noise is generated by sampling random numbers from the Gaussian distribution and adding them to the pixel values of the corresponding frames with the intensity of the noise determined by the mean and standard deviation of the Gaussian distribution.

Salt & Pepper noise adds scattered white and black pixels throughout the frames of the video independently resembling grains of salt and pepper on an image. After the pixels are randomly selected, their values are either set to maximum (white) or minimum (black).

(2) Blur Attacks. Noise attacks are a type of video frame manipulation technique where blurring effects are applied to frames independently to reduce the sharpness or clarity of the frame. Three types of blur attacks are tested in this work the Gaussian blur, Average blur, and Median blur.

Gaussian Blur is used to blur a frame of a video by applying a by a Gaussian function. Mathematically, applying a Gaussian blur to an image is the same as convolving the image with a Gaussian function or a Gaussian kernel. It applies the weighted average of neighboring pixels to each pixel in an image or video frame. The weights are determined by a Gaussian distribution centered around the pixel being blurred and the amount of blurring is controlled by adjusting the standard deviation parameter of the Gaussian distribution. The result is a softening effect that reduces high-frequency details and creates a smoother appearance.

Average Blur is a technique that replaces each pixel with the average value of its neighboring pixels. The size of the neighborhood, often represented by a kernel or window, determines the extent of blurring. It smooths out details and reduces sharpness, giving a more diffuse appearance to the video.

Lastly the Median Blur is a technique that replaces each pixel with the median value of its neighboring pixels. The median is determined from the intensity values within the kernel window and its effective in preserving edges and reducing noise.

(3) Enhance Attacks. Noise attacks are a type of video frame manipulation technique where certain aspects of the frame are altered to be enhanced or modified. The two enhance attacks tested in this work are the histogram equalization and the gamma correction.

Histogram Equalization is a technique used to adjust the contrast, especially when the image is represented by a narrow range of intensity values and brightness of an image or video by redistributing the pixel intensity values. Transforms the intensity distribution of the frame to make it more uniformly spread across the entire range. This redistribution increases the overall contrast and enhances details in the frames of the video and allows for areas of lower local contrast to gain a higher contrast.

Gamma Correction is a technique used to adjust the brightness and contrast of an image or video by applying a non-linear transformation to the pixel intensity values. It compensates for the nonlinear behavior of display devices and adjusts the perceived luminance. Gamma correction involves raising the pixel intensity values to a power exponent. The value of the exponent determines the strength of the correction. A gamma value less than 1 darkens the image or video, while a gamma value greater than 1 brightens it.

(4) Compression Attacks. Compression attacks are techniques used to manipulate to degrade the visual quality of individual frames within a video. These attacks target specific frames and intentionally modifying the content of specific frames before compression. This can include altering the pixel values, introducing noise, or applying filters to distort the visual information can impact the overall viewing experience.

(5) Crop Attacks. Crop attacks on video frames involve intentionally manipulating the video frames by cropping or removing portions of the frame's content. These attacks aim to alter the composition or focus of the video frames, potentially distorting the intended visual narrative or manipulating the viewer's perception. There are many types of crop attacks but in our case, we will evaluate content removal attack where a portion of the video frame is cropped.

5.3 Experimental Design

In this section we will discuss the techniques we will use to evaluate the proposed watermarking algorithm. More specifically the fidelity and imperceptibility of the watermark will be evaluated in the output video and the robustness of the watermark when frames of the video have been they have subjected to attacks. The evaluation results that will be presented were done by partitioning the frames into 8×8 blocks, and the integers used as watermarks were integers that their binary representation needs 4 bits, [8,15] meaning that each block was partitioned into 9×9 cells. The series of integers consisted of 64 integers in range [8,15].

5.3.1 Measuring Fidelity Imperceptibility

When measuring fidelity or imperceptibility in videos, you need to assess the visual quality of the videos to determine how well the watermarking algorithm preserves the original content. For that reason, we will use the following two metric techniques to evaluate the fidelity and the imperceptibility of the watermark.

(1) Peak Signal-to-Noise Ratio (PSNR): PSNR is a widely used metric to measure the distortion between the original and watermarked videos in decibels (dB). It calculates the ratio of the peak signal power to the mean square error (MSE) between the original and watermarked video. Higher PSNR values indicate better fidelity. In this work we will calculate the mean PSNR of all the frames of the two videos.

$$\overline{PSNR} = \frac{\sum_{i=0}^{num\ of\ frames} PSNR_i}{num\ of\ frames}$$

where $PSNR_i$ is:

$$PSNR = 20 \cdot \log_{10}\left(\frac{255}{\sqrt{MSE}}\right)$$

$$MSE = \frac{1}{N \cdot M} \sum_{n=1}^{N} \sum_{m=1}^{M} \| original(n, m) - watermarked(n, m) \|^2$$

Original and watermarked refer to the frames of the original and watermarked videos respectively of size N, M.

(2) Structural Similarity Index (SSIM): SSIM is a perceptual metric that measures the similarity of structures between the original and watermarked videos. It considers luminance (L), contrast (C), and structural information (S). SSIM values range from 0 to 1, with higher values indicating better fidelity. In this work we will calculate the mean PSNR of all the frames of the two videos.

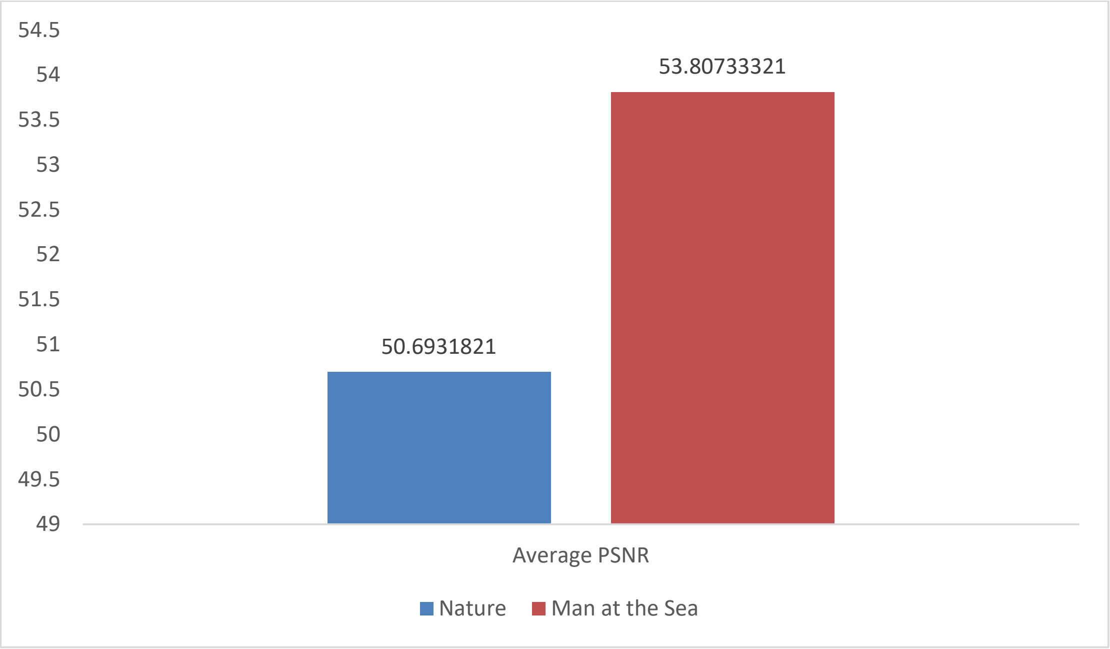
$$SSIM(Original\,, Watermarked) = L(Original\,, Watermarked) \cdot C(Original\,, Watermarked) \cdot S(Original\,, Watermarked)$$

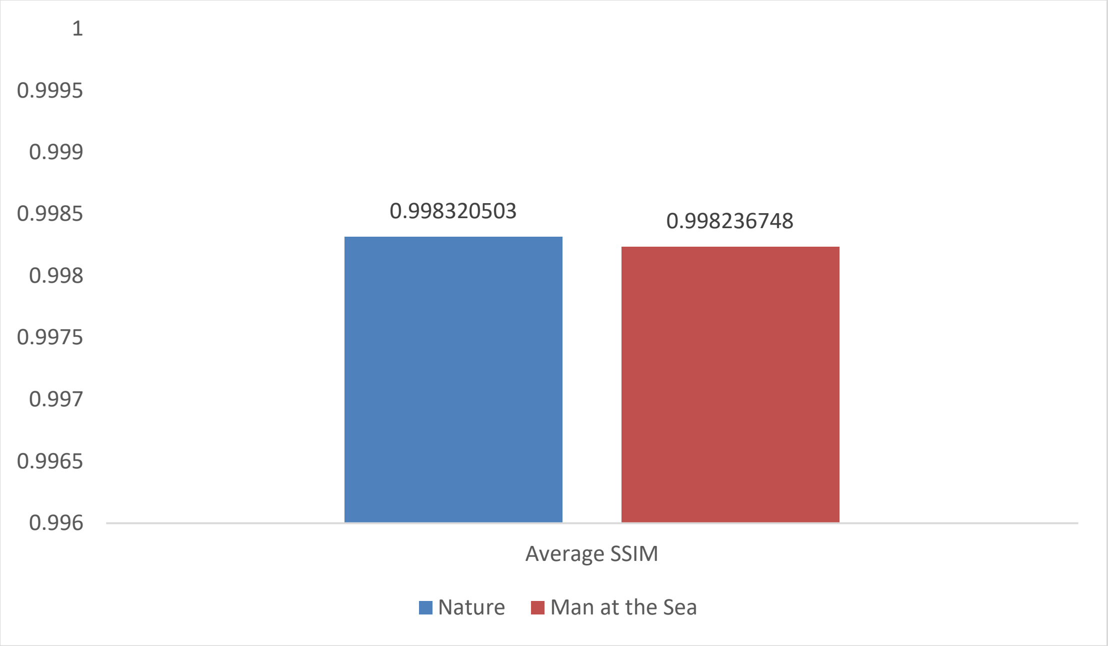
3.2.1 Robustness

In the context of video watermarking, robustness refers to the ability of a watermarking algorithm to withstand various attacks or transformations without significant degradation or loss of the embedded watermark. Evaluating the robustness of a video watermarking algorithm involves evaluating its resistance against intentional or unintentional modifications to the watermarked video. To evaluate robustness in this method we will compute the extraction rate of the watermarks we got after attacking the watermarked video with the attacks described in 5.2.

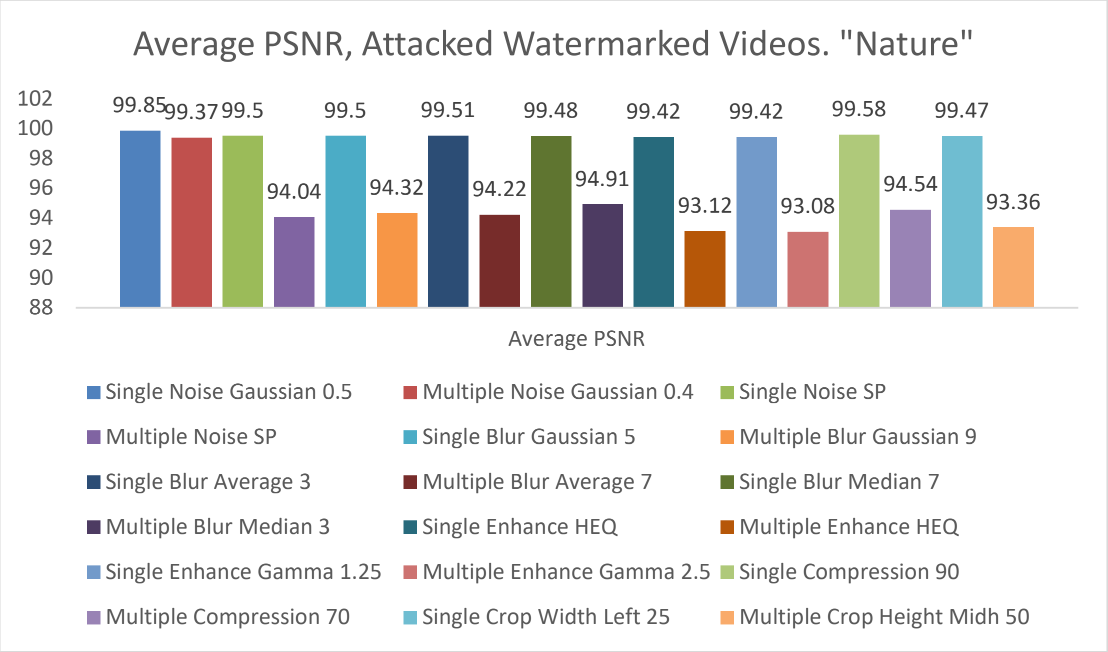
5.4 Video Fidelity Results

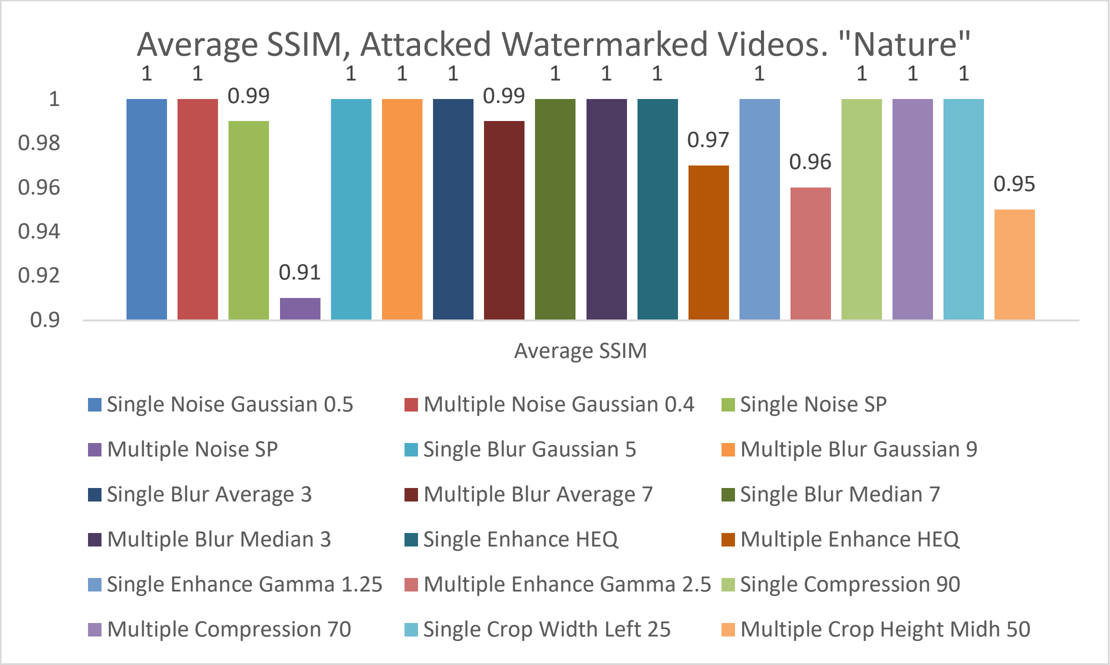
<u>Watermarked Video Results</u>

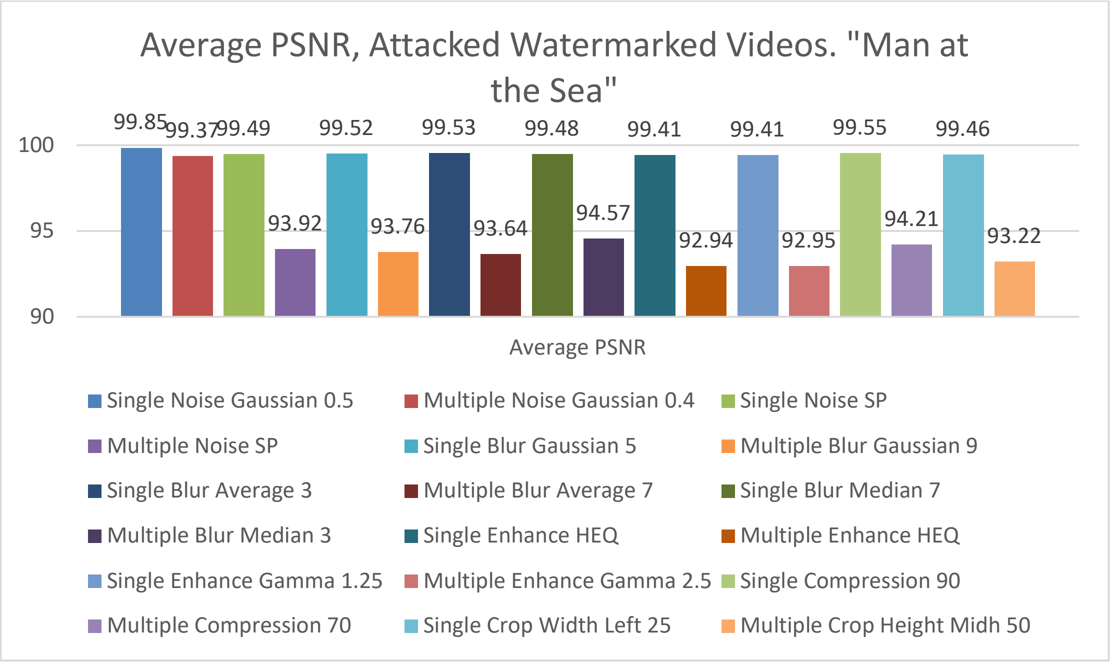


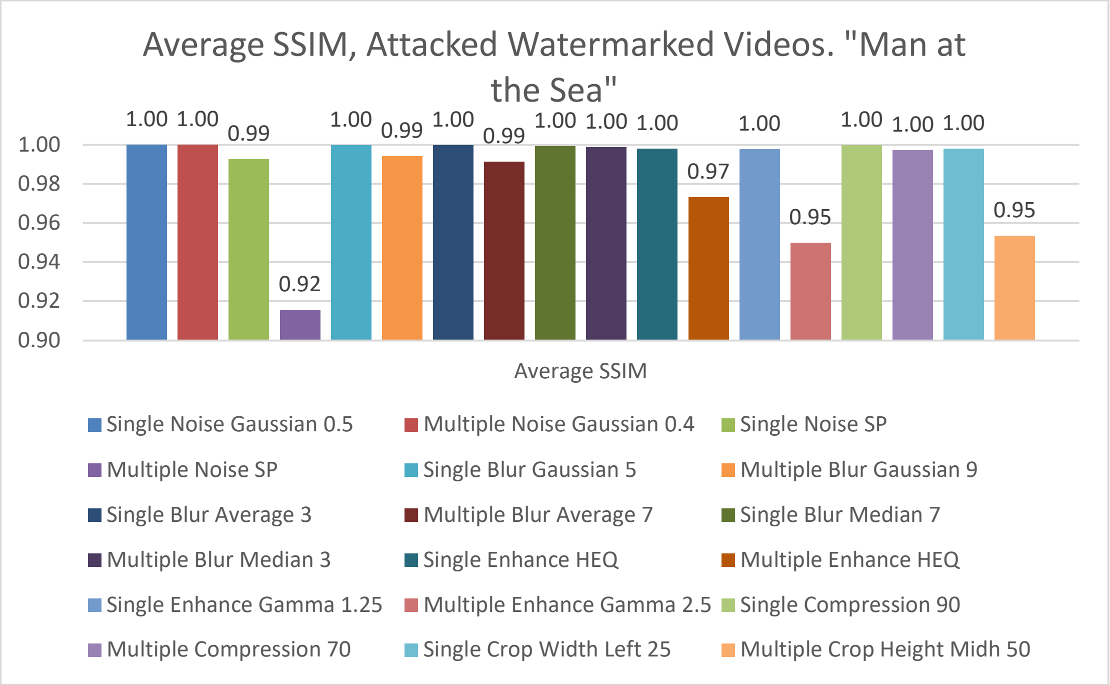


Attacked Videos Results




Average SSIM, Attacked Watermarked Videos. "Nature"
1
0.98
0.96
0.94
0.92
0.9
1
1
0.99
0.91
1
1
1
0.99
1
1
1
0.97
1
0.96
1
1
1
0.95
Average SSIM
Single Noise Gaussian 0.5
Multiple Noise Gaussian 0.4
Single Noise SP
Multiple Noise SP
Single Blur Gaussian 5
Multiple Blur Gaussian 9
Single Blur Average 3
Multiple Blur Average 7
Single Blur Median 7
Multiple Blur Median 3
Single Enhance HEQ
Multiple Enhance HEQ
Single Enhance Gamma 1.25
Multiple Enhance Gamma 2.5
Single Compression 90
Multiple Compression 70
Single Crop Width Left 25
Multiple Crop Height Midh 50


Average PSNR, Attacked Watermarked Videos. "Man at the Sea"
100
95
90
99.85
99.37
99.49
93.92
99.52
93.76
99.53
93.64
99.48
94.57
99.41
92.94
99.41
92.95
99.55
94.21
99.46
93.22
Average PSNR
Single Noise Gaussian 0.5
Multiple Noise Gaussian 0.4
Single Noise SP
Multiple Noise SP
Single Blur Gaussian 5
Multiple Blur Gaussian 9
Single Blur Average 3
Multiple Blur Average 7
Single Blur Median 7
Multiple Blur Median 3
Single Enhance HEQ
Multiple Enhance HEQ
Single Enhance Gamma 1.25
Multiple Enhance Gamma 2.5
Single Compression 90
Multiple Compression 70
Single Crop Width Left 25
Multiple Crop Height Midh 50



5.5 Video Authenticity Verification Results

5.5.1 Frame-by-Frame Verification Results

The following two tables show the authenticity verification results. During this evaluation, the extraction of the SIPs used during the embedding process was attempted. The results show the extraction rate of the integers for frames that have not been subjected to an attack in the fifth column and the extraction rate for attacked frames in the seventh column. On the left of the extraction rates are shown the number of frames in each case.

			#UF	$\bar{S}$ Extract (%) [UF]	#AF	$\bar{S}$ Extract (%) [AF]	Total
Nature			125	0.9985	-	-	
Noise Attacked Videos	Gaussian	Single Attacked Frame	124	0.9985	1	1.0000	0.999
		Multiple Attacked Frames	113	0.9986	12	0.9974	0.998
	S&P	Single Attacked Frame	124	0.9985	1	0.2188	0.992
		Multiple Attacked Frames	113	0.9985	12	0.1563	0.914
Blur Attacked Videos	Gaussian	Single Attacked Frame	124	0.9985	1	0.2031	0.992
		Multiple Attacked Frames	113	0.9986	12	0.2188	0.921
	Average	Single Attacked Frame	124	0.9985	1	0.2344	0.992
		Multiple Attacked Frames	113	0.9985	12	0.2135	0.920
	Median	Single Attacked Frame	124	0.9985	1	0.1875	0.992
		Multiple Attacked Frames	113	0.9983	12	0.2031	0.919
Enhance Attacked Videos	HEQ	Single Attacked Frame	124	0.9985	1	0.9844	0.998
		Multiple Attacked Frames	113	0.9985	12	0.9831	0.997
	Gamma	Single Attacked Frame	124	0.9985	1	0.8906	0.998
		Multiple Attacked Frames	113	0.9983	12	0.7708	0.976
Compressed Videos		Single Attacked Frame (90)	124	0.9985	1	0.25	0.993
		Multiple Attacked Frames (70)	113	0.9985	12	0.1966	0.918
Cropped Videos		Single Attacked Frame (25)	124	0.9985	1	0.7656	0.997
		Multiple Attacked Frames (50)	113	0.9986	12	0.5456	0.953

			#UF	$\bar{S}$ Extract (%) [UF]	#AF	$\bar{S}$ Extract (%) [AF]	Total
Man at the Sea			125	1.0000	-	-	
Noise Attacked Videos	Gaussian	Single Attacked Frame	124	1.0000	1	1.0000	1.000
		Multiple Attacked Frames	113	1.0000	12	1.0000	1.000
	S&P	Single Attacked Frame	124	1.0000	1	0.1875	0.994
		Multiple Attacked Frames	113	1.0000	12	0.1758	0.918
Blur Attacked Videos	Gaussian	Single Attacked Frame	124	1.0000	1	0.1875	0.994
		Multiple Attacked Frames	113	1.0000	12	0.2031	0.920
	Average	Single Attacked Frame	124	1.0000	1	0.2344	0.994
		Multiple Attacked Frames	113	1.0000	12	0.2214	0.922
	Median	Single Attacked Frame	124	1.0000	1	0.1875	0.994
		Multiple Attacked Frames	113	1.0000	12	0.2305	0.923
Enhance Attacked Videos	HEQ	Single Attacked Frame	124	1.0000	1	0.9375	1.000
		Multiple Attacked Frames	113	1.0000	12	0.9818	0.998
	Gamma	Single Attacked Frame	124	1.0000	1	0.8281	0.999
		Multiple Attacked Frames	113	1.0000	12	0.6784	0.968
Compressed Videos		Single Attacked Frame (90)	124	1.0000	1	0.2188	0.994
		Multiple Attacked Frames (70)	113	1.0000	12	0.2161	0.922
Cropped Videos		Single Attacked Frame (25)	124	1.0000	1	0.7656	0.998
		Multiple Attacked Frames (50)	113	1.0000	12	0.5143	0.951

5.5.2 Integrity Preservation Results

			Nature	Man at the Sea
Watermarked Video			Hash of Frames and Hash Extracted from Audio Match	Hash of Frames and Hash Extracted from Audio Match
Noise Attacked Videos	Gaussian	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
	Salt & Pepper	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
Blur Attacked Videos	Gaussian	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
	Average	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
	Median	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
Enhance Attacked Videos	Histogram Equalization	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
	Gamma	Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
Compressed Videos		Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match
Cropped Videos		Single Attacked Frame	Hashes do **not** Match	Hashes do **not** Match
		Multiple Attacked Frames	Hashes do **not** Match	Hashes do **not** Match

5.5 Discussion over the Exhibited Results

After the presentation of the evaluation of the watermarked method will be discussed the results in this section. The fidelity and imperceptibility results as already discussed were computed using the PSNR and SSIM values. For the verification of the video authenticity, we will focus on the number of the successfully extracted integers from the cells compared with the expected number of extracted integers, which is 64 integers in every frame. Lastly, to verify the preservation of the integrity in the frames of the video we compare the hash of the frames with the hash that was hidden within the audio of the video.

In the first image in video fidelity results are presented the average PSNR values of all the frames that were computed for the watermarked videos in comparison with the original video. Typical "good" PSNR values are considered values above 30dB and in this case the results for both videos are above 50dB. Similarly, good quality videos generally produce SSIM values above 0.97-0.98 and in our case the results are above 0.998 as shown in the second image in video fidelity results. From those results indicate that the watermark has not distort the frames of the video and the algorithm produces high quality videos and faithful copies of the original.

Next, we present the results for attacked videos where we either attack a single or 10% of the frames of the video. The results for average PSNR and SSIM show that attacked videos maintain high average quality which as expected is getting as more frames are subjected to some type of attack. The video authenticity results show that the watermarking information can extracted extremely successfully from unaltered frames (UM) as can be proved by the average extraction rate for unaltered frame ($\bar{S}$ Extraction (%) UM). As for the altered frames that have subjected to an attack the results very. The watermark proves to be extremely robust against Gaussian Noise attacks, Histogram Equalization and Gamma attacks and can successfully extract much of the series of integers used as watermarks if not all of them. On the other hand, extraction rate results for other attacks like Blurring attacks are not that great but unless every frame of the video is attacked the decoding phase of the algorithm can extract the necessary information to prove ownership of the video.

Lastly, the integrity results show that the algorithm can accurately detect tampering in the frames of the video and fails to recognize any attacked video as authentic, while successfully identifying only the watermarked video as authentic by comparing and matching the hash the frames with the hash embedded in the audio.

Even though we can make general conclusions about the results, we can find out more about this method by comparing the results of this method with the results of other proposed methods. In the cells of the results of this work are depicted the evaluation results while in the other cells is depicted the range of values in the experimental results of the respective works.

	This Work	[12]	[13]	[14]	[15]	[16]
Range of PSNR values	50.6931821, 53.80733321	[28.2, 44.373]	[35.97, 49.3]	[36, 50.93]	[46.81, 47.76]	[42.26, 52.76]

	This Work	[12]	[13]	[14]	[15]	[16]
Range of SSIM values	0.998236748 0.998320503	[0.869, 0.986]	[0.9899, 0.9998]	[0.98, 0.99]	[0.9978, 0.9993]	[0.9926, 0.9978]

As we can observe the proposed method archives extremely satisfactory results regarding the quality of the watermarked video when the PSNR and SSIM values are compared with the values of similar works in video watermarking.

6. Conclusion

6.1 Concluding Remarks

In this work an invisible watermarking technique for videos that embeds encoded integers as 2D representations of self-inverting permutations was presented that intends to prove the authenticity of the video. Given the block size and the “class” of integers that will be encoded as self-inverting permutations, the algorithm partitions each frame in the spatial domain and creates a random series of integers to embed each one of them into a block of the frames of the video. The positions within the block are defined by marked positions in the 2D representation of the permutation and are marked in the frequency domain by adding a value in the magnitude of the cell. Another part of the authentication process is the verification of the integrity if the video frames. For that purpose, the frames are encrypted using a cryptographic function which then is embedded in the audio of the video.

The method was evaluated for the quality of the watermarked video and was tested by attacking the some of its frames to test the robustness and test if the integrity is affected. The results were positive proving the imperceptibility of the watermark with metrics that test the similarity to the original video, with great extraction rate of the integers from the watermarked video and accurate results about the integrity. The fidelity and imperceptibility were also tested against other methods of video watermarking and proved to produce great results even exceeding many of them.

6.2 Potentials and Limitations

The proposed method offers many benefits that makes it appeal for further investigation take advantage to its full potential. First, the extraction of the watermark needs little information to be extracted. A couple of files that either indicate position for the extracted algorithm to locate marked areas or to validate the extracted integers are all that the algorithm needs and especially not needing the original video at any point of the extraction process. It offers great extraction rate even if a frame or many frames of the video have been attacked. The imperceptibility of the watermark in the frequency domain, especially with the partition of the frames and the use of 2D representations of the self-inverting permutations is another major advantage of this method and makes it extremely difficult even if an attacker that suspects the existence of a watermark to accurately locate the marked areas without the information that the user has. Lastly, another advantage this method has is its ability to detect tampering in frames using the blockchain based method. This promising feature in combination with the frame partition can be used to detect changes within blocks and inform the user which blocks have sustained modifications rather than simply inform us that a change or changes have happened.

This method may offer many advantages but has some limitations too. First limitation is the trade-off between robustness and imperceptibility makes it unclear what the input value that will be used to modify the magnitude in the frequency domain should be. Robustness needs the embedded watermark to remain intact against various attacks meaning that, in general terms, a high value should be utilized. On the other hand, imperceptibility needs the less amount of embedded information, so the frames will not be distorted. Another limitation is the partition of the frames. Partition into small consistent and detailed areas referred as fine-grained partitioning provides high precision in locating regions but that precise localization comes with higher computational cost and provide low capacity on the size of the embedded information. Coarse-grained partitioning on the other hand creates larger areas which gives us the ability embed information with higher capacity, reduces the computational cost but limits the ability to accurately locate areas.

6.3 Future Research

The results achieved through the evaluation of the proposed watermarking scheme show the capabilities of the method providing videos of high quality and robustness. But further analysis could be done to improve its capabilities and potentially improve the robustness against attacks that the method failed to give satisfactory results like blurring attacks. Moreover, modifications could be done in the architecture of the algorithm to explore further capabilities, for example tampering detection and localization within the frames. Further, the proposed technique can be extended for multiple watermarks for real time applications.

References

[1] Gwenaël Doërr, Jean-Luc Dugelay, A guide tour of video watermarking.

[2] Robust Image Watermarking Theories and Techniques: A Review, Hai Tao, Li Chongmin, Jasni Mohamad Zain, Ahmed N. Abdalla.

[3] Digital Image Watermarking Techniques: A Review, Mahbuba Begum and Mohammad Shorif Uddin.

[4] A survey of color image watermarking: State-of-the-art and research directions, Dhiran Kumar Mahto, A.K. Singh.

[5] Watermarking Images in the Frequency Domain by Exploiting Self-Inverting Permutations, Maria Chroni, Angelos Fylakis, Stavros D. Nikolopoulos.

[6] A Repetitive Watermarking Scheme for Digital Images based on Self-Inverting Permutations, Maria Chroni, Stavros D. Nikolopoulos, Iosif Polenakis, Vasileios Vouronikos.

[7] A discrete wavelet transform and singular value decomposition-based digital video watermark method, Qingliang Liu, Shuguo Yang, Jing Liu, Pengcheng Xiong, Mengchu Zhou.

[8] New Rapid and Robust Color Image Watermarking Technique in Spatial Domain, Qingtang Su, Decheng Liu, Zihan Yuan, Gang Wang, Xiaofeng Zhang, Beijing Chen, And Tao Yao.

[9] A robust blind medical image watermarking approach for telemedicine applications, Fares Kahlessenane, Amine Khaldi, Redouane Kafi, Salah Euschi.

[10] Hash-based image watermarking technique for tamper detection and localization, Muzamil Hussan, Shabir A. Parah, Aiman Jan, G. J. Quresh.

[11] A Detailed look of Audio Steganography Techniques using LSB and Genetic Algorithm Approach, Gunjan Nehru, Puja Dhar.

[12] Manish K Thakur, Vikas Saxena, J. P. Gupta, A Performance Analysis of Objective Video Quality Metrics for Digital Video Watermarking.

[13] Sibaji Gaj, Aditya Kanetkar, Arijit Sur, And Prabin Kumar Bora, Drift-Compensated Robust Watermarking Algorithm for H.265/HEVC Video Stream.

[14] Vincent Adul and Elijah Mwangi, A Robust Video Watermarking Approach based on a shybrid SVD/DWT Technique.

[15] Antonio Cedillo-Hernandez, Manuel Cedillo-Hernandez, Mireya Garcia-Vazquez, Mariko Nakano-Miyatake, Hector Perez-Meana, Alejandro Ramirez-Acosta, transcoding resilient video watermarking scheme based on spatio-temporal HVS and DCT.

[16] Milad Jafari Barani, Peyman Ayubi, Milad Yousefi Valandar, Behzad Yosefnezhad Irani, A blind video watermarking algorithm robust to lossy video compression attacks based on generalized Newton complex map and contourlet transform.