Image Watermarking: Review and Attacks, Final report

Ayush Anand(B20CS082), Aaditya Baranwal(B20EE001)

Abstract

Image Watermarking is the process of inserting copyright data inside an image through which it is possible for content creators to claim the ownership of a specific image. In this paper, we review some popular techniques used to insert watermarks in images and then explore attacks that could be carried out on them. As this is an intermediate progress submission, it is not complete yet. The code used for this paper is available on https://github.com/iamayushanand/Image_Watermarking.

1. Introduction

Image Watermarking is the process of inserting copyright data inside an image through which it is possible for content creators to claim the ownership of a specific image. Usually, the data which is to be inserted inside the image is not restricted to a specific modality and is passed as input in the form of bytes. In this section, we formalize the notion of Image Watermarking.

1.1. Embedding

Let the image in which we insert the embedding be called I. This will also be called the **Cover Image**. The watermark data that must be inserted inside the Cover image is called W. W could have any modality. However, depending on the type of watermarking scheme, specific modalities are more convenient to handle(such as a visual watermarking scheme would prefer W to represent an image).
Apart from these, specific watermarking techniques also require a **secret key** K. This secret key is shared between the sender and the receiver. The receiver can use this key K to retrieve the watermark and confirm the authenticity of the image sent.
The output of the embedding scheme is the image D. We formalize this as

$$D = Enc(I, W, K)$$

1.2. Extraction

The Process of extracting the watermark from the image D considers the parameters D, I (optionally) and K (optionally). More formally, it is formalized as:

$$W^{'} = Dec(D, I, K)$$

. The watermark is accepted if the correlation(measured using some metric) exceeds some threshold T. i.e

$$corr(W, W^{'}) \geq T$$

This formalism allows us to create an abstract class for watermarking algorithms(to be found in the watermarking.py file)

2. Dataset

We use a sample of the COCO dataset with 128 images and 100 custom images which were captured by us for evaluating our algorithms. We add a fixed watermarked data to the images and then compare them.
We convert strings into bytes whenever possible if the data to be inserted is a string. Similarly in case of ADD algorithm we fixed a particular image and inserted it across all the dataset.
To make sure our comparitive study is valid we ensured that the number of bytes added to each image for each algorithm stays the same.

3. Contributions

This project was developed by Ayush Anand(B20CS082) and Aaditya Baranwal(B20EE001).
Our contributions are as follows:

- Ayush Anand: LSB algorithm, DCT algorithm and Visual Watermarking attack.
- Aaditya Baranwal: Additive algorithm and DWT algorithm.

4. Metrics Implemented

The metrics play an important role in assessing the Quality of watermarking. In general, a watermarking scheme is evaluated based on certain parameters such as:

1. **Imperceptibility**: The watermarked image and the original image should not have a lot of visible differences between them.

2. **Robustness**: Transformations (such as the addition of noise) should not destroy the watermarked data present in the image.

3. **Capacity**: How much watermark data can be hidden inside an image?

4.1. Imperceptibility

To measure the imperceptibility of the images, we use metrics such as MSE, PSNR and MSSIM.

4.1.1 MSE

MSE gives a rough idea of the difference between the two images. It, however, does not take into account the structural similarity of the images in any way and is based on the sum of squared pixel intensity differences.

$$MSE = \frac{\Sigma_x \Sigma_y (A[x][y] - B[x][y])^2}{MN}$$

here M, N represents the dimension of the images.

4.1.2 PSNR

PSNR stands for peak signal-to-noise ratio. It is calculated as

$$PSNR = 10 \log \frac{L^2}{MSE}$$

Here L represents the maximum intensity in the image. For our purpose, it is 255.
It is customary in the field of image processing to convert MSE to PSNR values. Higher PSNR implies more similarity between the cover image I and the watermarked image D.

4.1.3 MSSIM

MSSIM stands for Mean Structural Similarity Index Measure. It is intended to provide a measure of reliability between two images considering the visual human system. This metric ranges from 0 to 1. Higher values represent more similarity between watermarked image and the original image.
first the images are broken into B blocks each. Then the MSSIM is calculated as:

$$MSSIM = \frac{1}{B} \Sigma_B l(x,y) c(x,y) s(x,y)$$

$$l(x,y) = \frac{2\mu_x\mu_y + (K_1 L)^2}{(\mu_x)^2 + (\mu_y)^2 + (K_1 L)^2}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + (K_2 L)^2}{(\sigma_x)^2 + (\sigma_y)^2 + (K_2 L)^2}$$

$$s(x,y) = \frac{2\sigma_{xy} + (K_2 L)^2}{(\sigma_x)^2 + (\sigma_y)^2 + (K_2 L)^2}$$

K_1 and K_2 are small constants used to avoid the case of a 0 denominator, while L represents the maximum intensity(in our case, 255).

4.2. Robustness

To check the robustness of the algorithms, we first augment the watermarked images by adding different kinds of noise to them. Then we compute the BER (Bit error rate) of the extracted watermark and of the original watermark.

$$BER = \frac{WB * 100}{TB}$$

where WB represents the erroneous bits and TB represents the total bits.

4.3. Capacity

The capacity of an algorithm is inherent to its working. As such, it does not need a metric and can be calculated based on the algorithm alone.

4.4. Implementation

These metrics can be implemented in the repository under the metrics.py file.

5. Spatial Domain Watermarking

We implemented two spatial domain watermarking algorithms. The data is hidden inside the individual pixel intensity values in spatial domain watermarking.

5.1. LSB watermarking

LSB stands for least significant bits. The watermark data W here is represented in bits. Specific pixels of image I are chosen, and the LSB of that corresponding pixel is set to some corresponding bit in W.

5.1.1 Pros and Cons

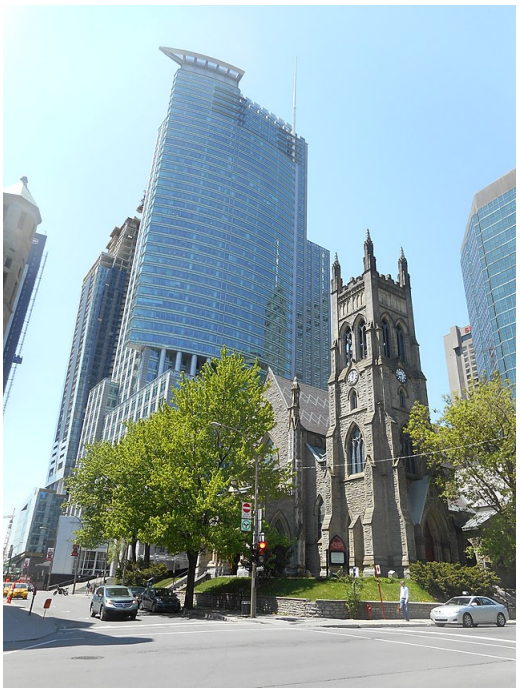
1. It is very simple to implement.

2. Is not robust. Simple noise addition can destroy the watermark

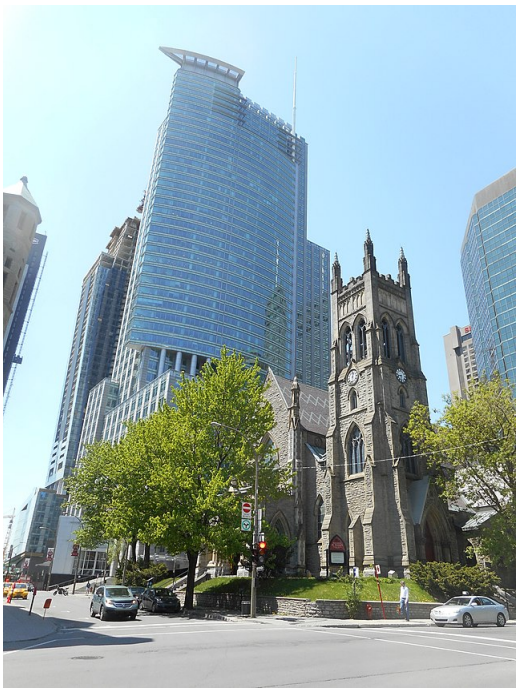
3. Very high imperceptibility due to manipulation only at the least bit.

5.1.2 Result

The visual outputs are shown in figure 1.



(a) Original



(b) Watermarked

Figure 1. LSB algorithm

Metric	Value
MSE	0.0009
PSNR	62.18
MSSIM	0.989

Table 1. Results.

5.2. Additive watermarking

Additive Watermarking is a technique to add a hidden digital watermark to a digital image or other multimedia content to protect its ownership or authenticity. The watermark is embedded into the image by adding it to the pixel values so that it is not easily visible to the human eye but can be detected and extracted by a computer program or algorithm. The added watermark can prove ownership or detect unauthorized copying or tampering with the original content.

5.2.1 Detailed Explanation

A detailed explanation is that Additive watermarking is embedding a digital watermark into an image or other multimedia content by adding it to the original pixel values using a key or secret code. Mathematically, this can be expressed as:

$$I_{watermarked} = I_{original} + \alpha * W$$

where $I_{watermarked}$ is the watermarked image, $I_{original}$ is the original image, W is the watermark to be embedded, alpha is the scaling factor or key used to control the strength of the watermark, and the addition and multiplication are performed on a pixel-by-pixel basis. The resulting watermarked image is visually similar to the original image but contains hidden information that can be extracted using an appropriate algorithm or software. Additive watermarking is a widespread technique due to its simplicity and effectiveness and is widely used in applications such as copyright protection, content authentication, and digital forensics.

5.2.2 Pros and Cons

1. The main advantage of additive watermarking is its simplicity and effectiveness, as it does not require complex mathematical operations or transformations and can be easily implemented using simple arithmetic operations. Additionally, additive watermarking is robust to familiar image processing operations such as compression, scaling, and cropping, which may affect the image's spatial and frequency domain coefficients.

2. However, the main disadvantage of additive watermarking is that the watermark may be easily visible or detectable if the scaling factor or key used to embed the watermark is too strong or if the watermark

is not properly masked or randomized. Additionally, additive watermarking may be vulnerable to more advanced attacks, such as signal processing techniques or image manipulation tools designed to remove or modify the watermark.

5.2.3 Result

Metric	Value
MSE	0.0020
PSNR	58.72
MSSIM	0.985

Table 2. Results.

The visual outputs are shown in figure 1.

6. Frequency Domain Watermarking

We implemented two Frequency domain watermarking algorithms. Frequency domain watermarking involves embedding a watermark in the frequency domain of an image. This is done by performing a mathematical transformation on the image, which converts it into a new representation where the high-frequency information, which is responsible for details and edges, is separated from the low-frequency information, which contains the overall structure and smoothness of the image. The watermark is then embedded in the high-frequency coefficients, which are less noticeable to the human eye.

6.1. DCT Watermarking

DCT stands for Discrete Cosine Transform. It is a transformation which decomposes a signal as a sum of **cosines**. This choice is deliberate, as it has been noticed that fewer cosines are required to represent an image compared to sines.
Instead of hiding the data in the pixel values, it is instead hidden in the coefficients of the cosines found using the D.C.T.
The bits of the message are usually hidden in the LSB of the coefficients. As is the case in our implementation of DCT watermarking.

6.1.1 Pros and Cons

1. Slightly tricky to implement
2. Imperceptibility is not as good as LSB
3. Robust to most transformations.
4. There is a tint on the image depending on which channel the data is hidden in.

6.1.2 Results

Metric	Value
MSE	19.813
PSNR	36.01
MSSIM	0.979

Table 3. Results.

The visual results are shown in Figure 3.

6.2. DWT Watermarking

DWT Watermarking involves embedding a watermark in an image using the discrete wavelet transform. This involves breaking down the image into smaller components and embedding the watermark in one or more. This technique is used in digital image processing to protect copyright or verify the authenticity of an image.

6.2.1 Detailed Explanation

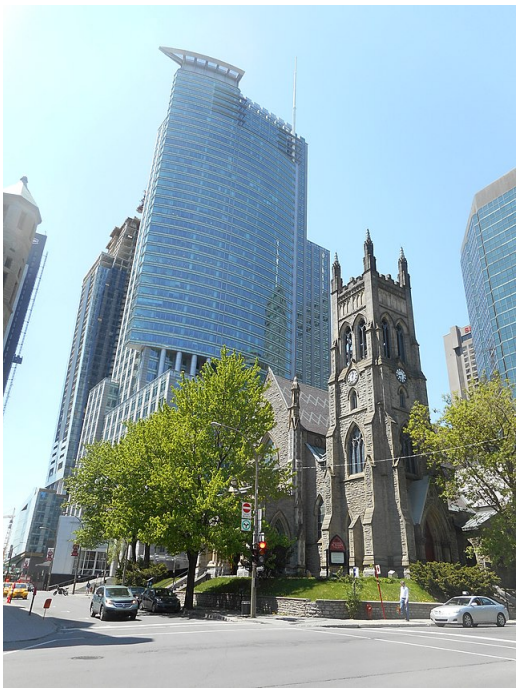
Mathematically, DWT watermarking involves using the discrete wavelet transform (DWT) to decompose the image into different frequency subbands, which are then used to embed the watermark. The DWT decomposes the image into four subbands: LL (low-low), LH (low-high), HL (high-low), and HH (high-high), where the LL subband contains the majority of the image energy, while the remaining subbands contain the high-frequency coefficients. The watermark is then embedded in one or more high-frequency subbands, less noticeable to the human eye. This technique is commonly used in digital image watermarking to provide robustness against various image processing attacks.

6.2.2 Pros and Cons

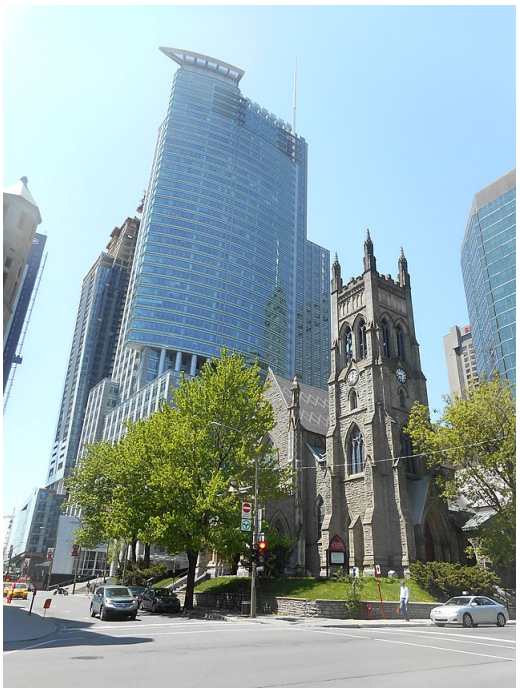
DWT Watermarking edges over other techniques under the following headings:

1. Multiresolution embedding: DWT watermarking is a multiresolution watermarking technique, meaning that the watermark is embedded in multiple levels of the DWT decomposition. This makes the watermark more robust to different image processing attacks, such as compression or cropping.

2. Perceptual masking: DWT watermarking considers the perceptual characteristics of the human visual system (HVS). The high-frequency subbands are less noticeable to the human eye, and the watermark can be embedded in these subbands without affecting the image's visual quality.

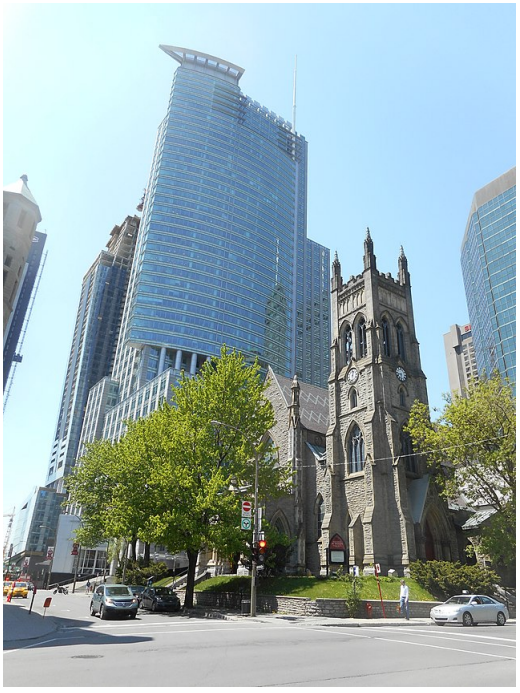


(a) Original

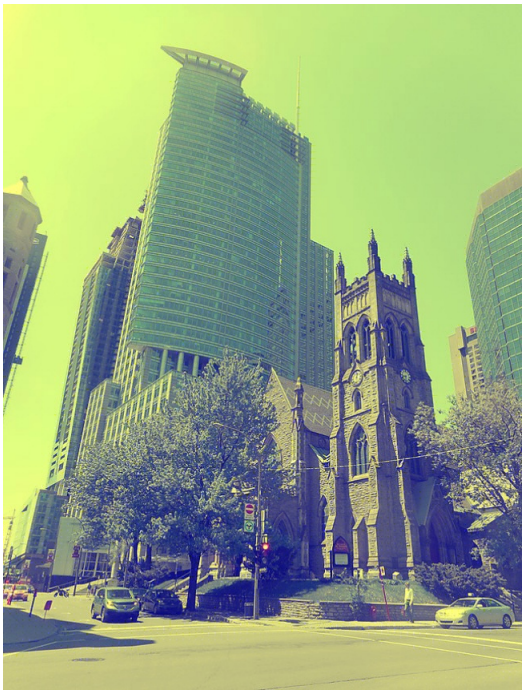


(b) Watermarked

Figure 2. ADD algorithm



(a) Original



(b) Watermarked

Figure 3. DCT algorithm

3. Robustness: DWT watermarking is more robust to various image processing attacks such as compression, filtering, and cropping. This is because the high-frequency subbands are less affected by these attacks

than the low-frequency subbands.

4. Security: DWT watermarking can also provide some security against unauthorized access to the watermark. The watermark can be encrypted using a secret key, and the decryption key can recover the original watermark. This makes it more difficult for an attacker to remove or modify the watermark without the decryption key.

Overall, DWT watermarking is a powerful technique for digital image watermarking that offers many advantages over other watermarking techniques.

The cons of DWT include its:

1. Sensitivity to image compression: DWT watermarking is sensitive to image compression techniques such as JPEG, which means that if an image is compressed using such techniques, the watermark may be lost or degraded.

2. Complexity: DWT watermarking algorithms can be complex, making them difficult and time-consuming to implement.

   Limited item capacity: DWT watermarking has a limited capacity for embedding information, which means that only a small amount of data can be embedded in an image using this technique.

6.2.3 Results

Metric	Value
MSE	16.129
PSNR	32.41
MSSIM	0.994

Table 4. Results.

7. Comparative

7.1. On 128 images of COCO Dataset

Methods	Additive	LSB	DCT	DWT
MSE	5.89	0.0009	19.813	0.1723
PSNR	34.89	62.18	36.01	55.76
MSSIM	0.963	0.985	0.979	0.999
BER	-	-	0.0034	0.0019

7.2. On Handmade Dataset of 100 Images

Methods	Additive	LSB	DCT	DWT
MSE	6.01	0.0012	21.342	0.1813
PSNR	33.97	61.76	37.23	55.49
MSSIM	0.973	0.981	0.980	0.997
BER	-	-	0.0032	0.0021

8. Visual Watermarking Attack

In Visual watermarking, the watermark W is hidden as a part of the image and visible to anyone. This is implemented simply by adding an overlay on the cover image I. Stock photos are an example of such watermarking.
We model a watermarked image J by

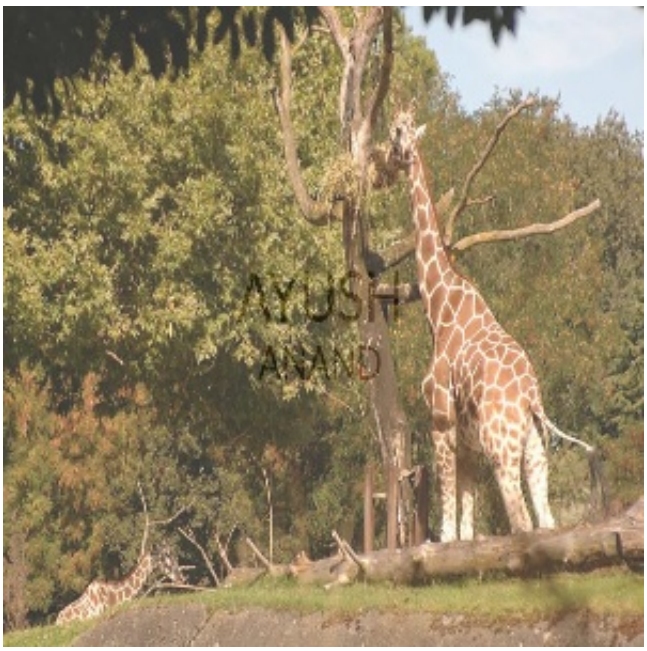
$$J = \alpha W + (1 - \alpha)I$$

Here α is the matting constant. We assume that we have a collection of images and they are constructed using the same matrix W and the same α. The paper https://ieeexplore.ieee.org/document/8100209, describes a method to remove watermarks even if they are shifted. However for our project we implement a toned down version of the method in the CVPR paper.
The original image can be recovered by

$$I = \frac{J - \alpha W}{1 - \alpha}$$

The attack has been implemented in the visual/removal_attack.py file
In some cases there is a whitish overlay on the image and it is unable to give proper removal of the watermark.
The visual results are shown on the next page.

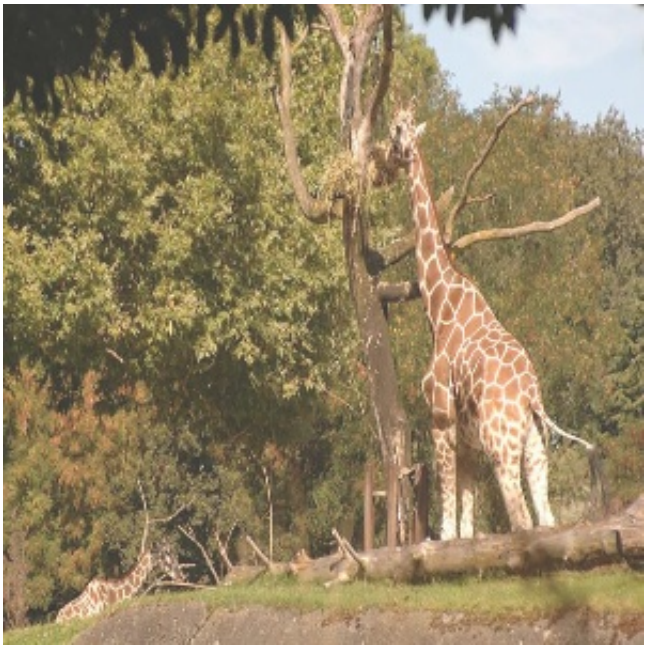
(a) Original



(b) Watermarked

Figure 4. Visual Watermarking

(a) Original



(b) After Watermark removal

Figure 5. Visual Watermarking