

# Invisible watermarking using Discrete Cosine Transform (DCT): A comprehensive exploration

Krishna Kumar Bohra <sup>1,\*</sup> and Vinod Kumar Soni <sup>2</sup>

<sup>1</sup> Faculty of Computer Science, Department of Computer Science, Lachoo Memorial College of Science and Technology (Autonomous), Jodhpur, Rajasthan, India.

<sup>2</sup> Faculty of Science, Department of Physics, Lachoo Memorial College of Science and Technology (Autonomous), Jodhpur, Rajasthan, India.

World Journal of Advanced Research and Reviews, 2024, 24(02), 2639–2645

Publication history: Received on 18 October 2024; revised on 25 November 2024; accepted on 28 November 2024

Article DOI: <https://doi.org/10.30574/wjarr.2024.24.2.3623>

## Abstract

Invisible watermarking is a technique used to protect digital media, like images and videos, by adding a secret mark (watermark) that cannot be seen. This helps prevent copying and ensures the content's authenticity. One popular method for invisible watermarking is called Discrete Cosine Transform (DCT), which works by changing the image into a form where small changes can be made without affecting its quality. DCT focuses on the frequency of the image, which helps hide the watermark in a way that it's not noticeable to the human eye. This technique is used in many areas like copyright protection and security. In this paper, we explain how DCT works, how to apply it using Python, and where it can be used. We also discuss the advantages and disadvantages of DCT-based watermarking compared to other methods, showing why it's a strong choice for protecting digital content.

**Keywords:** Invisible Watermarking; DCT-Based Watermarking; Inverse DCT; Python Implementation

## 1. Introduction

With the rapid increase in digital media, protecting intellectual property has become more important than ever. Invisible watermarking is an effective way to safeguard content, as it hides information within the media without affecting its quality. Unlike visible watermarks, which are easy to see, invisible watermarks are embedded in a way that cannot be detected by the human eye. These hidden marks can later be extracted to prove ownership or verify the authenticity of the content. One of the most used techniques for invisible watermarking is the Discrete Cosine Transform (DCT). DCT is widely used in image compression methods like JPEG because it can effectively reduce the size of the image while retaining important visual information. DCT works by transforming the image into frequency components, where the watermark can be embedded in a way that is not noticeable, even though the image's overall appearance remains unchanged. This paper discusses how DCT-based watermarking works, the steps involved in its implementation and provides a Python-based algorithm for embedding and extracting watermarks. It also looks at the advantages of using DCT for watermarking and how it compares to other methods in terms of robustness and quality preservation.

## 2. Understanding Invisible Watermarking

Invisible watermarking is a method of embedding hidden information, such as ownership details or authentication data, into digital media like images, videos, or audio without altering their perceptual quality. Unlike visible watermarks, these remain undetectable to human senses but can be retrieved using specific algorithms. The watermark is embedded

\* Corresponding author: Krishna Kumar Bohra

by subtly modifying the file's properties, such as pixel intensities in images or frequency components in audio. This technique ensures the original content appears unchanged while safeguarding intellectual property. Invisible watermarking is robust against modifications like compression or scaling, making it crucial for copyright protection, authentication, and tracking digital assets.

Key features of invisible watermarking include:

- **Imperceptibility:** The watermark should not reduce the media's quality. For example, a watermark added to an image should not change its appearance or make it blurry.
- **Robustness:** The watermark should survive common changes to the media, such as resizing, compression, or adding noise. For instance, even after a photo is compressed into a smaller file size, the watermark should still be detectable.
- **Security:** Only authorized users should be able to extract or detect the watermark. This ensures that only the rightful owner can prove their ownership of the media.

## 2.1. Application of Invisible watermarking

Invisible watermarking has a wide range of applications, particularly in digital media security and intellectual property protection. An example of this is embedding a hidden watermark in a digital image to prove ownership, even if the image is resized or shared online. Some key applications include:

- **Copyright protection:** Watermarks can be embedded in images, videos, and music to prove ownership and prevent unauthorized use or distribution. For example, artists can watermark their digital artwork to protect against theft.
- **Tamper detection:** Watermarks can be used to verify the integrity of digital files. If the watermark is altered or removed, it can indicate that the file has been tampered with, making it useful in legal or sensitive documents.
- **Digital rights management (DRM):** Watermarking helps enforce the rights of content creators by embedding tracking information, ensuring that only authorized users can access or distribute the content. For example, a movie streaming service may use watermarking to track illegal sharing of videos.
- **Authentication:** Watermarks can verify the authenticity of products, documents, or media, helping prevent counterfeit items. For example, luxury brands can use watermarks in digital images of their products to prove their authenticity.
- **Fingerprinting:** Watermarking can be used for digital fingerprinting, where each copy of a media file is uniquely marked to identify the original source. This is helpful in tracking and managing media distribution.
- **Secure communication:** Watermarks can be used to hide confidential messages in digital media, ensuring that only the intended recipients can detect and decode the hidden information.

## 2.2. Overview of Discrete Cosine Transform (DCT)

The Discrete Cosine Transform (DCT) plays a significant role in various digital signal processing applications. It can be broadly categorized into two types: Global DCT watermarking and Block-based DCT watermarking. As a frequency-domain transformation technique, DCT is recognized for its higher robustness against attacks compared to approaches that operate in the spatial (time) domain (1).

### 2.2.1. What is DCT?

Discrete Cosine Transform (DCT) is a mathematical method used to convert data from the spatial domain (where the data is represented as pixels, for example, in an image) to the frequency domain (where the data is represented by different frequency components).

Imagine that an image is made up of many tiny colored dots (pixels). Each of these dots has a certain value representing its color. The DCT takes this image data and breaks it down into a combination of different frequencies (like low, medium, or high frequencies). These frequencies describe the variations in pixel values across the image.

In simpler terms, the DCT allows us to represent the image using cosine functions that oscillate (or wiggle) at different speeds (frequencies). Most of the important information about the image (such as basic shapes or edges) is captured by low frequencies, while finer details or noise are captured by high frequencies.

This is useful because we can compress or hide information (such as a watermark) in the parts of the image where human eyes are less likely to notice changes (the high-frequency components).

Three types of coefficients can be chosen: (2)

- If the DC component is chosen, as in this paper, then there is a risk of changing the contrast of some blocks, as the DC is an average of luminance values.
- If the low or mid-range frequency coefficients are chosen, the image quality could be affected because the human eye is especially sensitive to these frequencies.
- If high frequencies are used, they are likely to be removed at compression time.

### 2.2.2. Properties of DCT

- **Energy Compaction:** This means that the DCT is very efficient at concentrating most of the important information (or energy) of the image into a small number of components, particularly the low-frequency ones.
- **Separability:** The DCT can be applied in a separable manner, which means we can first apply the DCT to the rows of an image and then apply it separately to the columns.
- **Real-Valued Output:** Unlike the Fourier Transform, which gives complex numbers (with both real and imaginary parts), the DCT produces real numbers only.

## 2.3. Mathematical Representation

For a 1D signal  $x(n)$  of length:

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot \cos \left[ \frac{\pi}{N} \left( n + \frac{1}{2} \right) k \right]$$

### Equation 1: Mathematical operation used in image processing and watermarking

The equation in the image represents the Discrete Cosine Transform (DCT), which is commonly used in invisible watermarking. For a 2D image, the DCT is applied to both rows and columns, enabling frequency-domain analysis (3). The formula computes the DCT coefficient  $X(k)$  for a signal  $x(n)$ , where:

- **N:** Total number of samples (or pixels in an image block).
- **$x(n)$ :** Pixel values of the signal (or image).
- **cos:** Basis function that determines how much of a particular frequency is present in the signal.
- **DCT Definition:** The equation transforms a signal  $x(n)$  (e.g., pixel values of an image) into its frequency components  $X(k)$ . It expresses the data as a sum of cosine functions at different frequencies.
- **Watermark Embedding:** In invisible watermarking, the watermark is embedded in the DCT coefficients (frequency components) of the digital file. By modifying certain coefficients (often higher-frequency ones), the watermark becomes imperceptible to the human eye but detectable algorithmically.

## 3. DCT-Based Invisible Watermarking

DCT-based watermarking involves embedding a hidden watermark into the frequency components of an image. This technique is used to ensure that the watermark remains invisible to the human eye, while still being detectable by algorithms designed to extract it. The main goal is to embed the watermark in a way that does not degrade the visual quality of the image but makes it robust enough to survive common image processing operations, like resizing, compression, or noise addition.

### 3.1. Steps in DCT-Based Watermarking

Here is a step-by-step explanation of the DCT-based watermarking process:

#### 3.1.1. Preprocessing:

Convert the watermark and host image into appropriate formats:

- The first step is to prepare both the watermark and the host image (the image we want to watermark).
- Typically, both the image and the watermark are converted to grayscale to simplify the process. This is because DCT-based watermarking usually works with intensity values (grayscale) rather than colour images, although the same principles can be extended to colour images.
- Example: Suppose we want to watermark an image of a landscape. The landscape image would first be converted into grayscale, and the watermark (like a logo or text) would also be in grayscale.

### 3.1.2. DCT Transformation:

Perform a block-wise DCT on the image:

- The next step is to apply the DCT to the image. Since DCT operates on blocks of data, the image is divided into smaller blocks, typically of size **8x8** pixels (1).
- The DCT is then applied to each block of pixels, converting the spatial information (pixel values) into frequency components.
- The DCT transforms the image from the spatial domain (where pixel values are defined in terms of position) to the frequency domain (where data is represented as a combination of various frequencies).
- **8x8 blocks** are commonly used for compatibility with JPEG compression standards, where each block's data is transformed separately.

#### Example

For an image block of size 8x8 pixels, DCT converts it from spatial pixel values to 64 frequency components (each representing a different frequency in the block). The lower frequencies represent the basic structure of the block (smooth regions), while higher frequencies represent finer details (edges, textures).

### 3.1.3. Watermark Embedding:

Embed the watermark into selected DCT coefficients

- Now, the watermark is embedded into the frequency components (DCT coefficients) of the image. The watermark is typically embedded into the **mid-frequency** components (4), as these frequencies are less noticeable to the human eye but still important for the image's overall structure.
- Low frequencies are avoided because they carry the essential information that defines the image's basic shape and modifying them might distort the image visibly. High frequencies, which represent finer details, are also avoided because they are often discarded during compression (5).
- The watermark is usually embedded by adjusting the amplitude of selected DCT coefficients based on a predefined pattern.

### 3.1.4. Inverse DCT:

Apply the inverse DCT to reconstruct the image:

- After embedding the watermark into the DCT coefficients, the image is transformed back from the frequency domain to the spatial domain using the **Inverse DCT (IDCT)** (6).
- This step reconstructs the image, which now includes the hidden watermark, but ideally, the watermark should not be visible to the human eye because it has been embedded in the frequency components where the human visual system is less sensitive.
- After embedding the watermark, we apply the IDCT to the modified coefficients, and the result is a new image that looks very similar to the original, but with the watermark embedded within the frequency components.

### 3.1.5. Extraction:

Extract the watermark by analysing the altered DCT coefficients:

- The final step is watermark extraction. To extract the watermark, we analyze the DCT coefficients of the watermarked image and compare them to the original image's DCT coefficients. The difference in these coefficients reveals the hidden watermark. (7)
- The watermark is retrieved by extracting the pattern that was embedded into the DCT coefficients. Depending on the method used for embedding, this can involve comparing the coefficients or using a specific algorithm to detect the changes made during the watermarking process.
- After extracting the DCT coefficients from the watermarked image, we subtract the original image's DCT coefficients (without watermark) from the modified image's coefficients. The difference reveals the watermark, which can then be decoded into its original form (like the logo or text).

## 4. Implementation of DCT-Based Watermarking in Python

Below is a Python implementation for embedding and extracting an invisible watermark using DCT.

### 4.1. Watermark Embedding

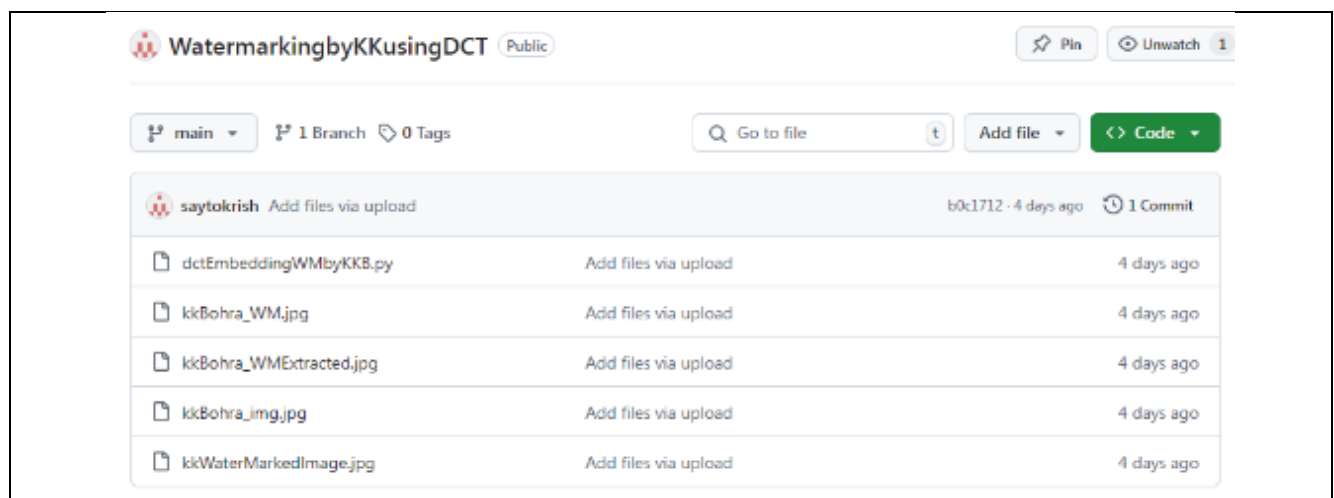
```
def embed_watermark(image_path, watermark_path, output_path, alpha=0.5):
    image = cv2.imread(image_path)
    watermark = cv2.imread(watermark_path, cv2.IMREAD_UNCHANGED)
    watermark = cv2.resize(watermark, (image.shape[1], image.shape[0]))
    if len(watermark.shape) == 2 or watermark.shape[2] == 1:
        watermark = cv2.cvtColor(watermark, cv2.COLOR_GRAY2BGR)
    watermarked_image = cv2.addWeighted(image, 1, watermark, alpha, 0)
    cv2.imwrite(output_path, watermarked_image)
    print(f"Watermarked image saved to {output_path}")
```

Figure 1 Embedding Watermark using Python

### 4.2. Watermark Extracting

```
def extract_watermark(watermarked_path, original_path, output_path, alpha=0.5):
    watermarked_image = cv2.imread(watermarked_path)
    original_image = cv2.imread(original_path)
    if watermarked_image.shape != original_image.shape:
        raise ValueError("Watermarked and original images must have the same dimensions.")
    watermark = (watermarked_image - original_image * (1 - alpha)) / alpha
    watermark = np.clip(watermark, 0, 255).astype(np.uint8)
    cv2.imwrite(output_path, watermark)
    print(f"Extracted watermark saved to {output_path}")
```

Figure 2 Embedding Watermark using Python



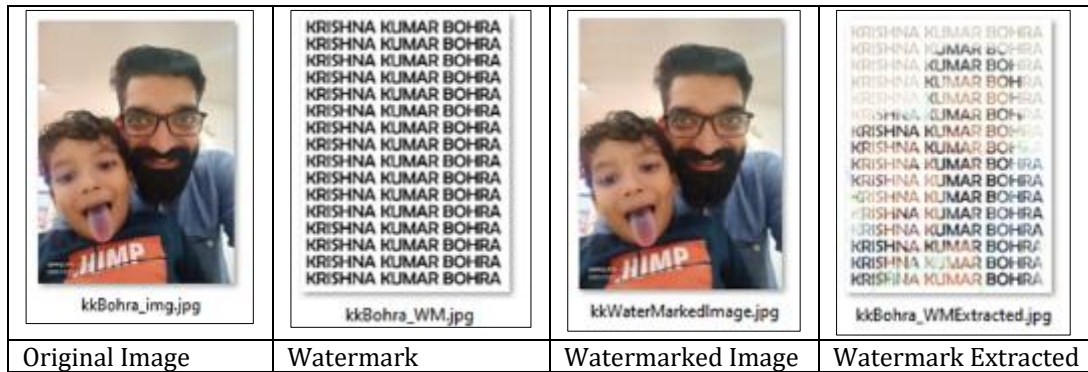
The screenshot shows a GitHub repository page for 'WatermarkingbyKKusingDCT' (Public). The repository is on the 'main' branch with 1 branch and 0 tags. It contains 5 files:

File Name	Commit Hash	Time	Commits
dctEmbeddingWMbyKKB.py	b0c1712	4 days ago	1 Commit
kkBohra_WM.jpg		4 days ago	
kkBohra_WMExtracted.jpg		4 days ago	
kkBohra_img.jpg		4 days ago	
kkWaterMarkedImage.jpg		4 days ago	

Figure 3 GitHub Link for Complete Code

Click here for Above complete code on GitHub [saytokrish/WatermarkingbyKKusingDCT](https://github.com/saytokrish/WatermarkingbyKKusingDCT)

## 5. Results



**Figure 4** Process/Result of Embedding and Extracting Watermark

The code provided performs two main operations:

- **Embedding a Watermark**
  - The function `embed_watermark` takes an image and a watermark, embeds the watermark into the image using alpha blending (with a specified strength of  $\alpha=0.01$ ), and saves the watermarked image.
  - **Input:** A color image (`kkBohra_img.jpg`) and a watermark image (`kkBohra_WM.jpg`).
  - **Output:** A new image (`kkWaterMarkedImage.jpg`) that contains the watermark embedded in it, with minimal distortion to the original image.
- **Extracting the Watermark:**
  - The function `extract_watermark` takes the watermarked image and the original image, then extracts the watermark by reverse engineering the blending process. The extracted watermark is saved as a new image (`kkWaterMarkedImage.jpg`).
  - **Input:** The watermarked image (`kkWaterMarkedImage.jpg`) and the original image (`kkBohra_img.jpg`).
  - **Output:** A new image (`kkBohra_WaterMarkedExtracted.jpg`) that contains the extracted watermark, ideally matching the original watermark (`kkBohra_WM.jpg`).

### 5.1. Expected Results

- **Watermarked Image (`kkWaterMarkedImage.jpg`):**
  - The image `kkWaterMarkedImage.jpg` will appear very similar to the original image `kkBohra_img.jpg`, but with the watermark subtly blended into it. Given that  $\alpha=0.01$  is used, the watermark should be almost invisible to the human eye, but detectable by an algorithm.
- **Extracted Watermark (`kkBohra_WaterMarkedExtracted.jpg`):**

The image `kkBohra_WaterMarkedExtracted.jpg` will ideally contain the watermark that was embedded in `kkWaterMarkedImage.jpg`. It may appear like the original watermark `kkBohra_WM.jpg` but with some possible distortions due to the blending process. Since alpha blending is not a perfect reversible operation (depending on the alpha strength and the image content), the extracted watermark may not match the original watermark perfectly but should still be recognizable.

## 6. Conclusion and Future Work

DCT-based invisible watermarking strikes a balance between imperceptibility and robustness, making it a preferred choice for many applications. Future advancements could integrate machine learning techniques, enhancing the security and robustness of watermarking systems.

Incorporating hybrid methods, such as combining DCT with DWT or cryptographic algorithms, can further bolster the capabilities of watermarking in a dynamic digital landscape.

- The **embedding process** effectively hides the watermark within the image without significant visual distortion when using a small alpha value (like  $\alpha=0.01$ ).



- The **extraction process** can recover the watermark, but due to the nature of alpha blending and potential changes in the image (like compression), the extracted watermark might not be an exact match to the original watermark.
- The results show that this approach is suitable for **invisible watermarking**, providing a method for embedding watermarks without noticeable visual changes. However, the **extraction quality** might be influenced by factors such as image compression or additional processing on the watermarked image. Therefore, while this method works well for basic watermarking applications, it may require additional techniques or modifications for higher robustness and accuracy, especially in challenging scenarios like lossy compression.

This exploration highlights the significance and practical implementation of DCT-based watermarking, demonstrating its relevance in protecting digital media.

---

## Compliance with ethical standards

### *Disclosure of conflict of interest*

No conflict of interest to be disclosed.



---

## References

- [1] Mansoori, Panthakkan. Robust Watermarking Technique based on DCT to Protect the Ownership of DubaiSat-1 Images against Attacks. IJCSNS International Journal of Computer Science and Network Security. 2021 June; 12(6).
- [2] Wang, Luo, Wang, Pan. A Hidden DCT-Based Invisible Watermarking Method for Low-Cost Hardware Implementations. MDPI Electronics. 2021 April.
- [3] Lidyawati , Ramadhan D, Jambola , Kristiana , Jayandanu. Digital watermarking image using three-level discrete wavelet transform under attacking noise. Bulletin of Electrical Engineering and Informatics. 2022; 11(1).
- [4] Al- H. Combined DWT-DCT digital image watermarking. Journal of Computer Science. .
- [5] Varghese , Hussain , Razak , Subash. A hybrid digital image watermarking scheme incorporating DWT, DFT, DCT, and SVD transformations. Journal of Engg. Research. 2021.
- [6] Zhao X, Zhang , Su , Vasana , Grishchenko , Kruegel , et al. Invisible ImageWatermarks Are Provably Removable Using Generative AI. In 38th Conference on Neural Information Processing Systems; 2024.
- [7] Begum. Analysis of Digital Image Watermarking Techniques through Hybrid Methods. Advances in Multimedia. 2020.

---

## Author's short biography

	<p><b>Dr Krishna Kumar Bohra</b>, PhD (Computer Science)</p> <p>I hold a doctoral degree in Computer Science and have been actively involved in research in the areas of Web Technologies and Computer Vision for over 10 years.</p>
	<p><b>Dr Vinod Kumar Soni</b>, PhD (Physics)</p> <p>I hold a doctoral degree in Physics and have been actively involved in research in the areas of Quantum, Spectroscopy and Nuclear Physics for over 10 years.</p>