Department of Electrical and Computer Engineering
University of Victoria
ELEC 483 - Digital Video Processing

Invisible watermarks

Report submitted on:	21 April, 2017
To:	Prof. P. Agathoklis
Names:	H. Emad (V00757795) T. Stephen (V00812021)

Abstract

This paper proposes a new image watermarking technique that encodes a plain-language phrase into an image DCT. We show how this method results in extremely small decreases to image quality and that embedding messages in the RGB layers results in less quality loss than embedding in the YCbCr layers. A method for two-party communication via watermarked images and a technique for blind watermark extraction are developed.

1 Introduction

Digital watermarking is the process of embedding information in an image to establish ownership and prevent unauthorized distribution. Traditional image watermarks that place translucent text or images over a source image must balance a degraded viewing experience with ease of removal. Watermarks that are placed in the corner of an image rarely cover up important image features but can be removed trivially through cropping. Prominent watermarks can discourage unauthorized image use (see Figure 1) but are impractical for use cases where a rights-holder wants to prevent dissemination of a high fidelity, minimally distorted image.

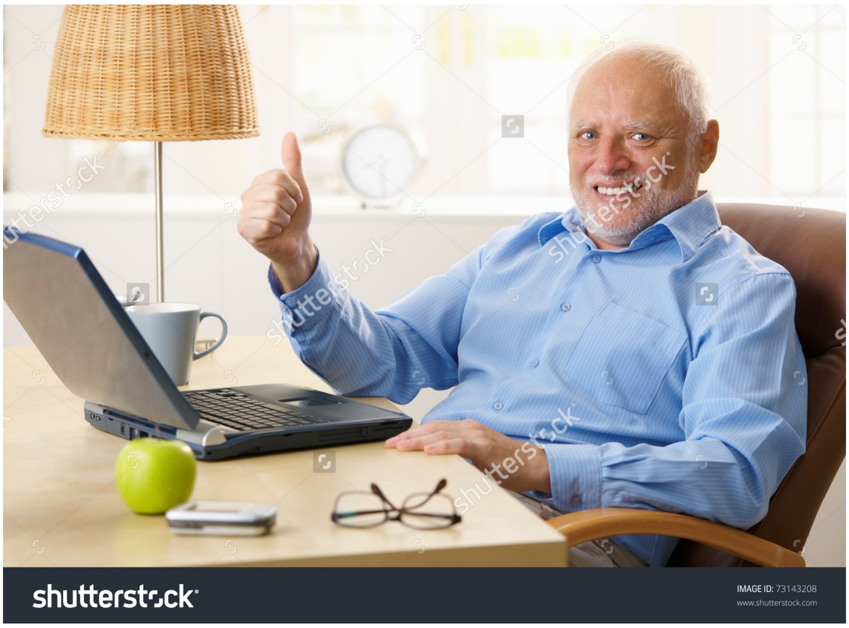


Figure 1: Stock image sites use overlay and footer watermarks to discourage unauthorized image use [1]

Image metadata (e.g. EXIF, IIPC, PLUS, Dublin Core) exists in the digital file header and leaves the source image unaltered. However, file metadata offers no protection against removal. cite needed An ideal digital watermark should be as inseparable from the source image as its namesake.

In this paper we demonstrate a procedure for watermarking digital files with almost no degradation of source image quality. In Section 3 we show how to take a plain language phrase and encode it in the image DCT blocks and then recover it. Section 4 shows that embedding the watermark in the frequency domain preserves image quality better than embedding a shorter message in the spacial domain. Section 5 considers the weaknesses of this procedure and outlines a method for two-party private communication via watermarked image exchange. It also proposes a method for blind watermark extraction.

2 Theory and analysis

Unless specifically designed to be otherwise, the largest frequency component of an image is its DC component. The DCT process typically yields a DC value over 1000. Altering this value by ±1 results in almost no change to image quality. This is the core principle at the heart of many compression algorithms.

cite needed

Relative image quality is determined by the Peak Signal-to-Noise Ratio (PSNR) value, representing the effect of noise from the $DCT \rightarrow IDCT \rightarrow DCT$ process and distortion from the watermark.

3 Implementation

The proof of concept was implemented in MATLAB with separate encode and decode stages. The encode stage takes an image file and a watermark phrase and outputs a watermark embedded image. The decode stage takes the watermarked and original images and outputs the extracted watermark message.

3.1 Encoding

Figure 2 shows the encoding process. The encode file prompts the user to supply a source image and message to embed. The message must be in basic alphanumeric format and the length is limited to 33 characters as a requirement for the BCH code.

The letters of the watermark message are associated with a number corresponding to its fre-

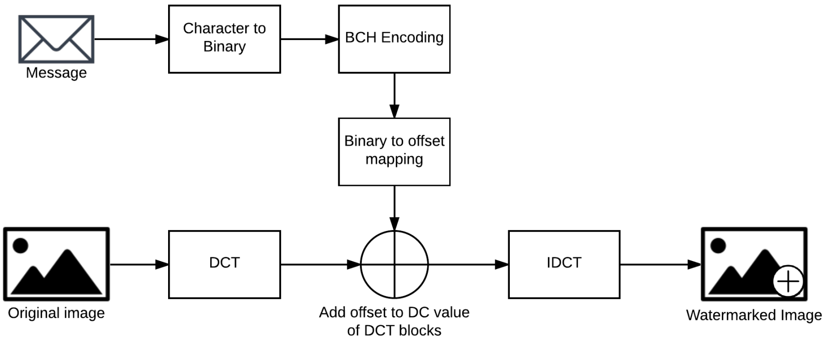


Figure 2: Watermark encoding process

quency in the English language.[1] This ensures that letters the most frequent letters will have more 0s, and thus a smaller distortion, when mapped onto the source image. The numbers are converted into 6-bit binary values[2] and form a binary sequence.

The *DCT* → *IDCT* → *cast to* `uint` process is *lossy*, primarily because of the downcast to `uint`. To minimize ambiguous downcasting we use an offset generation mapping $m, x \mapsto x'$ and recovery mapping $x, x' \mapsto m$ defined as:

$$x'_i = \begin{cases} x_i + 0, & m_i = 0 \\ x_i + 2, & m_i = 1 \end{cases} \qquad\qquad m_i = \begin{cases} 0, & x'_i - x_i \leq 0 \\ 1, & x'_i - x_i > 0 \end{cases}$$

where x is the original image, x' is the watermarked image and m is the BCH-coded message.

Despite this precaution, recovery errors occur with a rate $\approx 1\%$. In order to correct this, we apply a BCH code to the binary sequence to allow for error detection and correction. We use a BCH code with $n = 255$, $k = 199$ capable of correcting 7 errors (2.7451% error rate). To apply the BCH code, we pad the binary watermark sequence with 0s so it has 199 bits. The BCH encoder returns a sequence of 255 bits that can now be embedded into the image.

Next, we obtain the 8×8 block DCT of the image. Each binary value is mapped to the appropriate offset and added to the DC component of the DCT block, starting with the top left block. Once all 255 bits have been added to the DCT blocks, the image is reconstructed via Inverse DCT and written to an output file.

[1] See: https://en.wikipedia.org/wiki/Letter_frequency

[2] The inclusion of letters *and* numbers in the watermark character space pushes the total number of characters over 32, thus requiring 6 bits.

3.2 Decoding

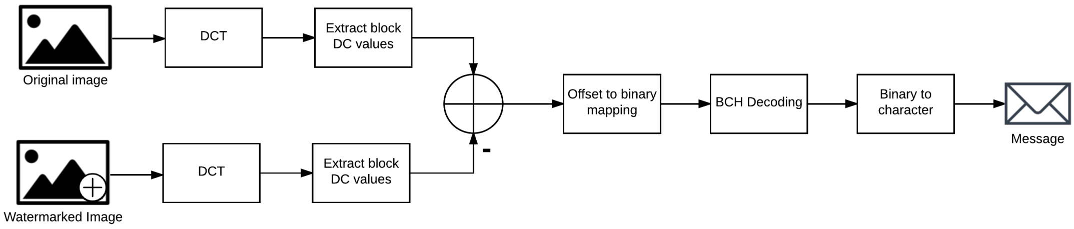


Figure 3: Watermark decoding process

Figure 3 shows the process for retrieving the watermark message from the image. This process is not "blind" since it requires the original image to extract the offset values in DCT DC coefficients. Moreover, the values of n, k for the BCH code and the structure of the character-to-binary map *must* be shared by the encode and decode processes.

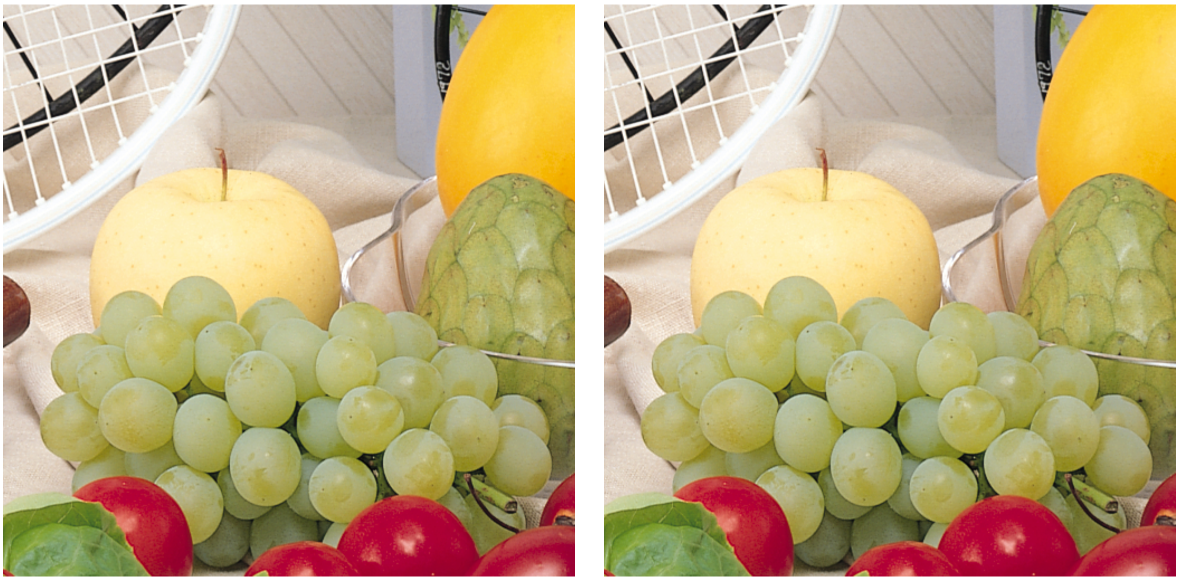
After recovering the binary sequence, 6-bit words are mapped to their original characters with padding characters mapping to `" "` empty space. The user is presented with the extracted watermark.

4 Data

Working versions of encoder and decoder scripts are available at https://github.com/trstephen/elec483. The scripts will embed and extract a text watermark from a provided grayscale or color image. Figure 4 shows the effect of adding a watermark to an image.

To evaluate the effects on image quality with spatial and frequency alterations, we compared the method described in Section 3 to one that embeds the binary sequence without the BCH code directly in the image grayscale values. This spatial embedding method produced a PSNR of 49.3924, significantly less than the PSNR of 58.7447 with our frequency embedding method. The frequency embedding method performs better because each message bit is encoded in a block of 64 pixels, rather than concentrated into single pixel. Additionally, the relative change in value from embedding a bit is smaller in the frequency domain.

We also attempted to determined the best "layer" to embed the watermark in a color image.



(a) Original

(b) Watermarked

Figure 4: A 22 character watermark has been embedded in the Y layer of the image

We used a couple test color images, each with RGB and YCbCr[3] encoded variants. The process in Section 3.1 was altered slightly to embed the watermark in a single layer and reconstruct the watermarked image using the watermarked layer and two unaltered layers. In addition to the PSNR, which measures errors in the image component values, we computed the Structural Similarity Index (SSIM) which measures changes to image luminance, contrast and structure. The SSIM is more predictive than PSNR for distortion as perceived by the human visual system [2].

The results in Table 1 show that embedding a watermark in any RGB layer causes less of a decrease in PSNR and SSIM than embedding it in any YCbCR layer.

The decrease in SSIM for watermarks embedded in the Y layer is not surprising since luminance is weighted in the SSIM value. This explains the better SSIM for RGB embedding since the luminance is a weighted sum of all RGB layers. However, the same decrease also occurs when the watermark is embedded in the chrominance layers even though chrominance changes are not measured directly by SSIM.

The effect of luminance changes on SSIM values does not explain why RGB embedding also has a better PSNR. This behavior is counterintuitive and warrants further research.

[3] Subsampled with 4:4:4 BT.601 standard.

Table 1: Effects on PSNR after embedding a watermark in a single layer

Source Image	Layer	PSNR	SSIM
peppersYCbCr	Y	58.91533	0.99997
	Cb	58.12800	0.99996
	Cr	58.95000	0.99996
peppersRGB	R	63.60164	0.99999
	G	63.74127	0.99999
	B	63.70763	0.99999
fruitsYCbCr	Y	58.88691	0.99988
	Cb	57.95389	0.99983
	Cr	59.07317	0.99987
fruitsRGB	R	63.88637	0.99995
	G	63.69708	0.99997
	B	63.59464	0.99995

5 Discussion

Troubles with errors and BCH? Feel like this is covered in Implementation...

We mention in Section 1 that the watermark cannot be removed from the image. This does not mean that the watermark cannot be destroyed. Since the extraction method relies on block-processing the images, any change to the block construction will drastically alter the resulting DCT. The watermark can be destroyed trivially by removing a single row of pixels! The security of this method relies on its ability to remain undetected.

If two parties wanted to communicate via exchange of watermarked images they would both need to have a copy of an original image to perform the message extraction. In order to avoid creating a suspicious communication pattern where they exchange and re-exchange the same image regularly the two parties should attempt to re-use one image for multiple exchanges. Posting “reaction images” and image memes on social media or other platforms would provide an excellent cover for sharing watermarked images since heavy image re-use is encouraged by the culture.

5.1 Further work

The motivated, row-dropping attacker problem notwithstanding, the need for an original image for decoding is a significant drawback for this method. Suppose instead we used the BCH encoded message to set the DC values of the blocks to specific values that could be extracted via modular division. The mappings would be

$$x'_i = \begin{cases} x_i + a_i \mid x_i + a_i \bmod 4 \cong 0, & m_i = 0 \\ x_i + a_i \mid x_i + a_i \bmod 4 \cong 2, & m_i = 1 \end{cases} \qquad m_i = \begin{cases} 0, & x'_i \bmod 4 \in \{0, 1\} \\ 1, & x'_i \bmod 4 \in \{2, 3\} \end{cases}$$

Observe that the recovery mapping is independent of the original image x. The modulus 4 is chosen avoid the casting errors described in Section 3.1 that would likely occur if a $0/1 \mapsto$ even/odd mapping were used.

6 Conclusion

This project succeeded in accomplishing its goal of demonstrating a procedure for watermarking digital files with almost no degradation of source image quality. By creating a proof of concept which embedded a digital phrase into digital image with minimal degradation to visual quality, and later extracted that same digital phrase with the help of BCH coding, this project demonstrated the potential and feasibility of this digital style of watermarking, in the real world.

The most obvious real world application of this technology is watermarking digital documents. Another potential use for this methodology is in establishing a digital chain of ownership for sensitive internal documents. Establishments will be able to keep an embedded digital record of the users who handled sensitive files. A record which they can trace back to the source of a leak or insurgence. Additionally, this form of embedding information in media can be used for covert communications between parties.

References

[1] StockLite. (2017). Happy senior man giving thumb up, sitting at desk using laptop computer at home, Shutterstock, [Online]. Available: https://www.shutterstock.com/image-photo/happy-senior-man-giving-thumb-up-73143208 (visited on 04/15/2017).

[2] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004, ISSN: 10577149. DOI: 10.1109/TIP.2003.819861.