A 8-BIT-GREY-LEVEL IMAGE EMBEDDING ITS 512 COLOR PALETTE

M. Chaumont and W. Puech

LIRMM Laboratory, UMR CNRS 5506, University of Montpellier II
161, rue Ada, 34392 MONTPELLIER CEDEX 05, FRANCE
phone: + (33)4.67.41.85.14, fax: + (33)4.67.41.85.00,
email: marc.chaumont@lirmm.fr, william.puech@lirmm.fr
web: www.lirmm.fr/icar/

ABSTRACT

In this paper, we propose a method to embed the high resolution color information of an image in its corresponding grey-level image. The objective of this work is to allow free access to the grey-level image and give high resolution color image access to secret key owners. This method is made of two steps which are the color image decomposition (in a grey-level image and its associated color information) and the color-information hiding. The main contribution of this paper is to consider the **decomposition into 512 colors** and to apply a reversible embedding of a bitplane plus a color palette. The resultant **watermarked grey-level** image is then **a 8-bit coded image**. With this watermarked image and the extraction secret key, a user is able to rebuild a color image composed of 512 colors. Considering the previous work, the two important consequences of this approach are the improvement of the security and a better color image quality.

1. INTRODUCTION

Previous work have been proposed for color protection but few of them both give a free access to low-quality images and a secure access to the same images with an higher quality. Our proposed solution is built on a data-hiding method. The image may be freely obtained but its high quality visualization requires a secret key. More precisely, in our solution, a grey-level image is freely accessible but only secret key owners may rebuild the color image. Our aim is thus to protect the color information by embedding this information in the grey level image.

Many work propose solutions to hide **information** by using the decomposition of a color image in an *index* image and a color palette [1, 2, 3]. The data-hiding may occur in the *index* image [1] or in the color palette [2, 3]. Nevertheless, none of those techniques tries to protect the color information by hiding the color palette in the *index* image. In our previous work, we present two different approaches in order to protect the color information by hiding the color palette in the *index* image [4, 5]. In [4] we propose to sort the colors of the color palette in order to get an *index* image which is near of the luminance of the original color image. In the same time, we get a color palette whose consecutive colors are close. In [5] the approach is completely different and relies on a function optimization of the global problem formulation. In those two previous work, the security and the color quality requirements where not addressed. The current paper gives new directions of analysis and improves the security and the quantized color image quality.

Other work such that [6, 7, 8] based on wavelet decomposition and sub-band substitution propose solutions to embed the color information in a grey-level image. Their areas are perceptive compression and image authentication for [6, 7] and image printing for [8]. Even if those techniques embed the color information, their approach and their purpose are clearly different from what our proposed approach.

The two new main contributions of the paper compared to the previous work of [4, 5] are:

- the used of 512 colors (*index* values belongs to [0, 511]; *index* image is thus a 9-bit coded image; and there are 512 colors in the palette) but a resultant grey-level image owning values belonging to [0 255]. The stored image (grey-level image) is then a classical 8-bit grey-level image and embed a color palette plus a bitplane.
- the compression of a bitplane and the color palette and the used of a high capacity reversible watermarking algorithm.

The general embedding scheme is given in Figure 1. The different steps are: the decomposition of a color image into an *index* image and a color palette (Section 2.1 and 2.2), the color palette coding and the coding of **a** bitplane (Section 2.3), and the watermarking of the 8-bit coded image built from the *index* image (Section 3). Note that the reversible-watermarking process consist in hiding the color palette plus a bitplane into a grey-level image. Finally the resultant grey-level image may be stored on a public website. The reversible extraction scheme, illustrated Figure 1, is the exact inverse of the embedding scheme.

2. IMAGE DECOMPOSITION PRINCIPLE AND MESSAGE CONSTRUCTION

2.1 Image decomposition

Reducing the color number of a color image is a classical quantization problem. The optimal solution, to extract the K colors, is obtained by solving:

$$\{P_{i,k}, C(k)\} = \arg \min_{P_{i,k}, C(k)} \sum_{i=1}^{N} \sum_{k=1}^{K} P_{i,k}.dist^2(I(i), C(k)), \quad (1)$$
$$and\ \forall i, \exists! k', \left[P_{i,k'} = 1\ and\ \forall k \neq k', P_{i,k} = 0\right],$$

where I is a color image of dimension N pixels, $C(k)$ is the k^{th} color of the research K colors, $dist$ is a distance function in the color space (L2 in the RGB color space), and $P_{i,k} \subset \{0,1\}$ is the membership value of pixel i to color k.

A well known solution to minimize the Equ. (1), and then to obtain the K colors, is to use the ISODATA k-mean clustering algorithm [9]. $P_{i,k}$ is defined as:

$$\forall i, \forall k, P_{i,k} = \begin{cases} 1 \text{ if } k = \arg\{\min_{\{k'\}} \min_{k' \in [1,K]} dist(I(i), C(k'))\}, \\ 0 \text{ otherwise,} \end{cases}$$

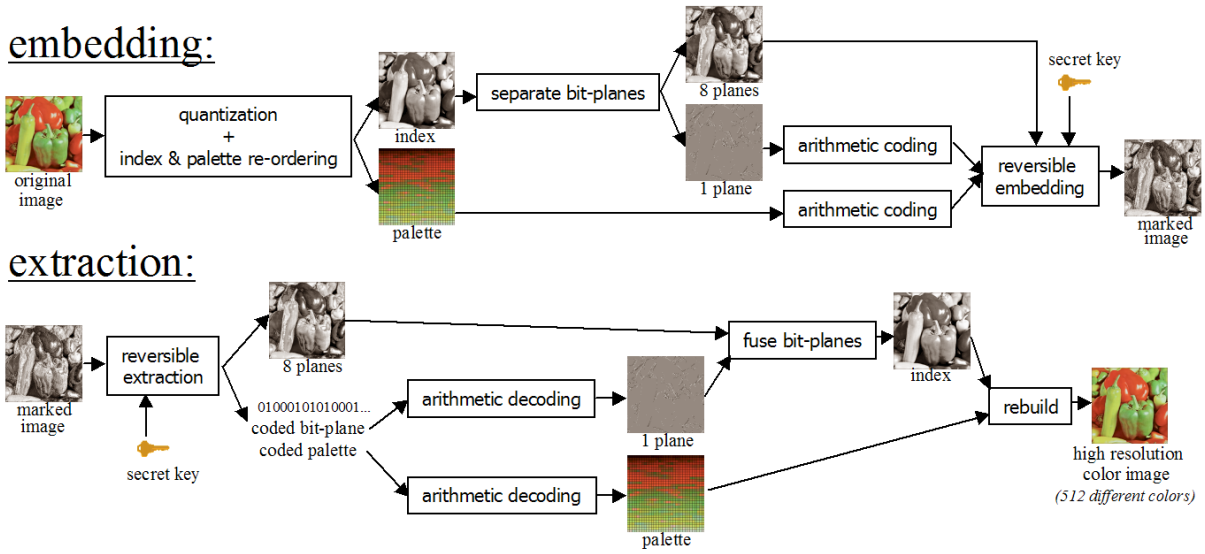


Figure 1: The general embedding/extraction scheme.

with $C(k) = \frac{\sum_{i=1}^{N} P_{i,k} \times I(i)}{\sum_{i=1}^{N} P_{i,k}}$.

Nevertheless, in our approach the K number is significant ($K = 512$). If we proceed with a classical k-mean algorithm, the number of colors extracted will often be below K. Indeed, it is the well known problem of *death classes*. Moreover, the k-mean algorithm is quite long in CPU time in comparison to non optimal but faster algorithms such that *octree color quantization algorithm* of Gervautz and Purgathofer [10], *Median Cut algorithm* [11]... To overcome those two problems ("death classes" and "CPU time"), we are using the *octree color quantization algorithm* as an initialization to the k-mean algorithm: $P_{i,k}$ are set from the result obtained with the *octree color quantization algorithm*.

2.2 Layer Running Algorithm

Once the color quantization has been processed, the obtained K color image could be represented by an *index image* and a color palette. The *index* image is noted down *Index* and is defined such that:

$$\forall i \in [1,N], Index(i) = \arg \max_{k \in [1,K]} P_{i,k}.$$

The color palette is noted down *Palette* and $\forall k \in [1,K], Palette(k) = C(k)$.

Our goal is to get an *index* image where each grey-level is not too far from the luminance of the original color image. A second weakest constraint is that in the color palette, two consecutive colors should be close. Thanks to the color quantization, we already own an *index* image and a color palette. Our problem is then to find a permutation function which permutes in the same time the values of the *index* image and the values of the color palette. The best permutation function Φ is found by solving:

$$\Phi = \arg\min_{\Phi} \sum_{\iota=1}^{N} dist^2(Y(\iota), \Phi(Index(\iota))) + \lambda \sum_{k=1}^{K-1} dist^2(Palette(\Phi^{-1}(k)), Palette(\Phi^{-1}(k+1))), \quad (2)$$

where Y is the luminance of the original color image, and λ is the Lagrangian value (λ is very small in order to privilege the equation first term). The Φ permutation function is a bijective function in $\mathbb{N}$ defined such that $\Phi : [1..K] \rightarrow [1..K]$.

In a first approximation, Equ. (2) is solved thanks to an heuristic algorithm: the *layer running algorithm* [4]. The aim of this algorithm is to find an ordering for the K colors such that consecutive colors are close and such that colors are ordered from the darkest to the lightest. This ordering defines for each k^{th} color a k' position which gives us the Φ function such that $\Phi(k) = k'$.

To find an ordering of the K colors, the algorithm runs the color space to build the ordered suite of colors, illustrated Figure 2. This running is obtained by jumping from color to color, into the color space, by choosing the closer color from the current one. The first color of this suite is chosen as the darkest one among the K colors. An additional constraint to this running is that we limit the color research to colors which are very close in luminance. This signify that the running in the color space is limited to a layer defined on luminance information. This *layer running algorithm* could then be seen as a kind of *3D spiral run* in the color space.

2.3 Message construction

Once the color image decomposition (into 512 colors) and its re-arrangement (*layer running algorithm*) have been proceeded, one bitplane of the *index* image is suppressed in order to create the cover image. The bitplane and the color palette are then compressed in order to be embeded into the

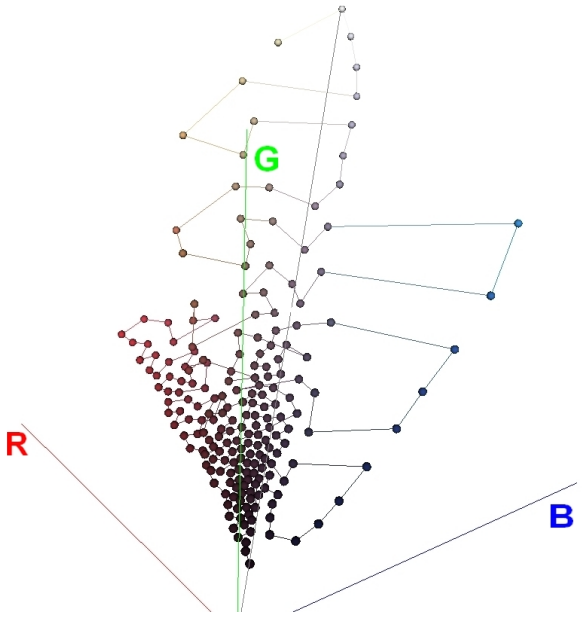


Figure 2: A view of the layer running in the RGB cube.

cover image (see Figure 1). The solution retained to compress the color palette is to statistically encode the error of prediction (differential encoding + arithmetic encoding) of each color belonging to the color palette.

The compression of the chosen bitplane is not enough efficient if directly proceeded. One have to modify the statistics in order to reduce the entropy (theoretical rate cost). Thus, the bitplane vector (noted bp) is transformed in a prediction error vector (noted e) owning three possible values : $\{-1,0,1\}$. Each bit $bp(i)$, $i \in [0,N]$, of the bitplane vector is then substituted by $e(i) \in \{-1,0,1\}$ before the lossless coding.

For each pixel $I(i)$ at position i, we first compute a prediction $Pred_{pixel}(i)$:

$$Pred_{pixel}(i) = \begin{cases} C \; if \; |A-B| < |A-C| \\ B \; otherwise, \end{cases} \tag{3}$$

with A, B and C the neighbors pixels of current pixel $I(i)$ as defined below:

previous line	A	B
current line	C	I(i)

This type of prediction is used in the Differential Pulse Code Modulation (DPCM) approach. It essentially take into account the edges in order to predict more cleverly than with a simple mean. Note that some more sophisticated predictions may be explored in particularly by looking at lossless compression technics. Nevertheless, this predictor is enough efficient for a first approach. Also note that for image border the prediction is proceeded with the available values.

The second step is to find the best prediction for the bit $bp(i)$ of the bitplane vector. At the coder and the decoder sides, the information of all the other bitplane are known. Thus, one have two possible prediction solutions: either predict a 0 knowing the other bitplanes, either predict a 1 knowing the other bitplanes. Knowing the prediction $Pred_{pixel}(i)$, one have to decide which is the best hypothesis: having a 0 in the bitplane (in this case the pixel value is noted $I_0(i)$) or having a 1 (in this case the pixel value is noted $I_1(i)$). This best hypothesis (best prediction) is chosen in accordance to the maximum of the two probability $p(Pred_{pixel}(i)|I_0(i))$ and $p(Pred_{pixel}(i)|I_1(i))$. This is a classical two hypothesis choice and with Gaussian pdf assumptions the best choice is:

$$Pred_{bit}(i) = \begin{cases} 0 \; if \; (Pred_{pixel}(i) - I_0(i))^2 < (Pred_{pixel}(i) - I_1(i))^2 \\ 1 \; otherwise. \end{cases} \tag{4}$$

The last step in order to obtain the e prediction error vector is to compute the prediction error (error values are either -1, 0 or 1):

$$e(i) = bp(i) - Pred_{bit}(i). \tag{5}$$

The prediction error vector may now be encoded and because of the good prediction behavior, the entropy is low and then the encoding cost is low.

3. REVERSIBLE WATERMARKING SCHEME

The algorithm used for the reversible embedding is one of the three most performant in term of embedding capacity. A brief description is given below; for more details see [12]. The algorithm lies on congruence computations. The congruence properties allow to define **three possible states** for a pixel: the *embedding* state (section 3.1), the *to-correct* state (section 3.2), the *original* state (section 3.3). Lets define the T transform which takes two integers x_1 and x_2 as input and return an integer:

$$\begin{aligned} T &: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N} \\ T(x_1,x_2) &= (n+1).x_1 - n.x_2. \end{aligned} \tag{6}$$

We will name the coding process, the act of embedding a message into an image and the decoding process, the act of extracting the message and rebuilding the original image. Lets now define the three possible states and the coding-decoding algorithms.

3.1 Embedding state

A pixel i in the *embedding* state is a pixel such that:

$$0 \leq T(I(i),I(i+1)) \;\; and \;\; T(I(i),I(i+1)) + n \leq L, \tag{7}$$

with I the original image (of N pixels size) whose grey-levels belongs to $[0,L]$. In the case of 8-bits images, L equals to 255. All pixels i which are in the *embedding* state:

1. are T transformed such that $I_T(i) = T(I(i),I(i+1))$, with $I_T(i)$ the resulting transformed pixel,
2. **must** then embed **a** coefficient w **belonging to** $[1,n]$ such that $I_w(i) = I_T(i) + w$.

After the embedding of a coefficient w belonging to $[1,n]$ we hold the congruence property (8). This congruence property allows to detect an *embedding* pixel during the decoding process.

$$(I_w(i) + n.I(i+1)) \; mod \; (n+1) \neq 0. \tag{8}$$

3.2 To-correct state

A pixel i in the *to-correct* state is a pixel such that the negation of equation (7) is true:

$$T(I(i),I(i+1)) < 0 \;\; or \;\; T(I(i),I(i+1)) + n > L.$$

All pixels that are in this *to-correct* state are then modified such that:

$$\begin{aligned} &c = (I(i) + n.I(i+1)) \ mod \ (n+1); \\ &if \ (I(i) - c) < 0 \ \ then \ \ c = -(n+1-c); \\ &I_w(i) = I(i) - c. \end{aligned} \tag{9}$$

The c **corrective codes** belong to $[-n, n]$ and are embedded (into the *embedding* pixels) in order to enable the reversibility of the *to-correct* pixels during the decoding process. Note that after the modification expressed by equation (9), pixel $I_w(i)$ checks the congruence property (10). This congruence property allows to detect a *to-correct* pixel at the reverting process.

$$(I_w(i) + n.I(i+1)) \ mod \ (n+1) = 0. \tag{10}$$

3.3 Original state

Given an image order scan for the coding, a pixel in the *original* state (i.e. unmodified pixel) should always be present just before a pixel in the *to-correct* state. For a top-bottom-left-right scan order, if a pixel i is in the *to-correct* state, then the pixel $i-1$ **must be** in the *original* state.

3.4 Coding and decoding algorithms

The **coding** process is composed of two steps:

1. classify each pixel in one of the three states: *embedding*, *to-correct*, *original*,
2. embed into the *embedding* pixels, the watermark made of corrective codes plus the message.

For the **decoding** process, the image scan order is inverted. The decoding process is also composed of two steps:

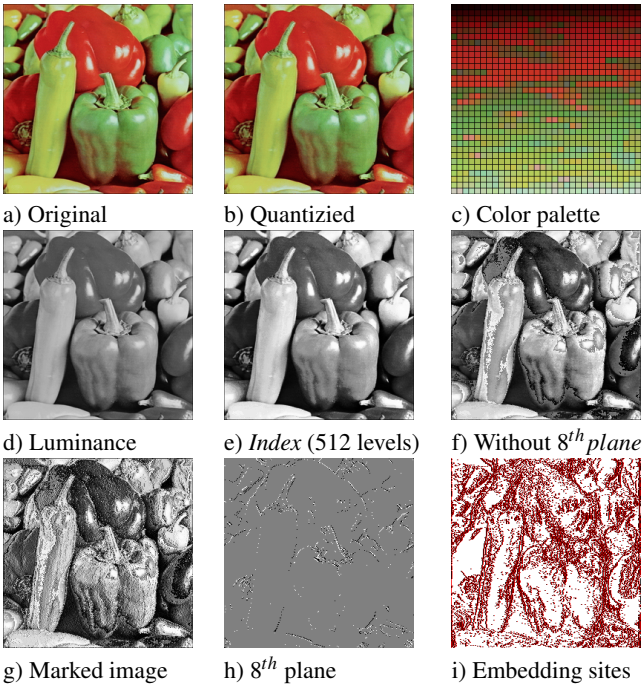
1. extract the watermark from the *embedding* pixels, revert (during the scan) all those pixels and localize the *to-correct* pixels,
2. from the extracted watermark retrieve the corrective codes and the message, and correct the *to-correct* pixels.

Note that the security is ensure by the secrecy of the scan order. The user secret key is used as a seed of a pseudo-random number generator. The obtained pseudo-random number sequence is used as scan order. Thus, no information about the message (color palette + bit plane) may be extract by an attacker. Moreover, an attack by colorization [13], which consist in retrieving semi-automatically (a small human intervention is necessary) the colors of each pixels, is difficult since the final grey-level image is of poor quality. Also note, that this reversible watermarking scheme is not robust to any signal processing but it is not a deficiency for this application.

4. RESULTS

We have applied our method on well known color images of size 256×256 pixels. For all the experiments, $n = 4$ for the reversible watermarking (see Equ. (6)). The results obtained show that the approach is efficient whatever the image type. In Figure 3, the main steps of our approach are comment for the *peppers* image.

After proceeding to the color decomposition of the *peppers* image (Figure 3.a) we obtain an *index* image whose index values belongs to $[0, 511]$ (Figure 3.e), and a color palette made of $K = 512$ colors (Figure 3.c). The quantized image



a) Original b) Quantizied c) Color palette
d) Luminance e) *Index* (512 levels) f) Without 8^{th} *plane*
g) Marked image h) 8^{th} plane i) Embedding sites

Figure 3: Steps of the color secured.

(rebuilt with the knowledge of the *index* image and the color palette) is given in Figure 3.b. The PSNR of this quantized image is 38.95 dB. The best PSNR obtained with a palette-based approach was 36.32 dB in [4]. The obtained **gain** with the approach proposed in this paper is superior to **2 dB** for the rebuilt color image.

The luminance image of the original color image is illustrated Figure 3.d. One could observe the good visual similarity between this luminance image and the *index* image. The *index* image is more contrasted but keeps its semantic intelligibility.

Once color image decomposition has been proceeded, the *index* image is separated into an 8-bit image (Figure 3.f) and a one bitplane image (a binary image). The chosen bitplane is the 8^{th} for *peppers* image. The binary image is then modified in order to obtain a ternary error prediction image of very low entropy. This error prediction image is given in Figure 3.h (in grey the 0 value, in black the -1 value, and in white the 1 value). The error prediction image and the color palette are then encoded with an arithmetic coder [14]. The embedded message is then made of the two bitstreams concatenations. Figure 3.i is a map giving the localization of the embedding site (white pixels) with the reversible watermarking approach of [12]. Figure 3.g shows the final 8-bit image embedding a color palette (of 512 colors) and a bitplane image (the 8^{th} bitplane of the *index* image).

The quality of the watermarked image (see Figure 3.g) is poor but as previously explain this is a good property. Indeed, this image yield difficult the colorization attack [13]. On the contrary, the rebuilt color image (knowing the secret key) gives a very good quality color image (PSNR = 38.95 dB).

An other result is shown for *barbara* 315×230 image in Figure 4. Figure 4.a is the original image, Figure 4.b is the quantized image with a PSNR of 38.75 dB and Figure 4.c

is the watermarked image. Note that the semantic content is preserved in this watermarked image.

For other images, PSNR are computed between the original color image and the rebuilt one, and given on Table 1, in order to compare with the previous best results obtained in [4]. For all the images, there is more than 2 dB gain with the proposed approach.

a) Original b) Quantized c) Watermarked

Figure 4: Example on barbara image.

Table 1: PSNR comparisons.

images	$\text{PSNR}_{(color\ image, rebuilt)}$	
	previous method [4]	proposed method
airplane	39.90 dB	42.96 dB
house	39.27 dB	41.67 dB
lena	38.63 dB	40.93 dB
peppers	36.32 dB	38.95 dB
baboon	33.31 dB	35.86 dB

5. CONCLUSION

In this paper, we have proposed a method to embed safely into a grey level image its color information. This method is based on a decomposition of a color image in an *index* image and a color palette with 512 different colors. The *index* image is then separated in a 8-bit image and a binary image. The binary image and the color palette are then reversibly embedded in the 8-bit image. The resultant watermarked image is still semantically understandable. Moreover, in comparison to previous palette-based approaches [4, 5], the quality of the rebuilt color image, knowing the key, is better, and the approach is more secure.

REFERENCES

[1] J. Fridrich, "A New Steganographic Method for Palette-Based Images," in *Proceedings of the IS&T PICS conference*, Apr. 1998.

[2] M.-Y. Wu, Y.-K. Ho, and J.-H. Lee, "An Iterative Method of Palette-Based Image Steganography," *Pattern Recognition Letters*, vol. 25, pp. 301–309, 2003.

[3] C.H. Tzeng, Z.F. Yang, and W.H. Tsai, "Adaptative Data Hiding in Palette Images by Color Ordering and Mapping With Security Protection," *IEEE Transaction on Communications*, vol. 52, no. 5, pp. 791–800, 2004.

[4] M. Chaumont and W. Puech, "A Fast and Efficient Method to Protect Color Images," in *IS&T/SPIE 19th Annual Symposium on Electronic Imaging, Visual Communications and Image Processing, VCIP2007, SPIE2007*, San Jose, California, USA, Jan. 2007, vol. 6508.

[5] M. Chaumont and W. Puech, "A Grey-Level Image Embedding its Color Palette," in *IEEE International Conference on Image Processing, ICIP'2007*, San Antonio, Texas, USA, Sept. 2007, vol. I, pp. 389–392.

[6] P. Campisi, D. Kundur, D. Hatzinakos, and A. Neri, "Compressive Data Hiding: An Unconventional Approach for Improved Color Image Coding," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 2, pp. 152–163, 2002.

[7] Y. Zhao, P. Campisi, and D. Kundur, "Dual Domain for Authentication and Compression of Cultural Heritage Images," *IEEE Transaction on Image Processing*, vol. 13, no. 3, pp. 430–448, 2004.

[8] R. de Queiroz and K. Braun, "Color to Gray and Back: Color Embedding Into Textured Gray Images," *IEEE Transaction on Image Processing*, vol. 15, no. 6, pp. 1464–1470, 2006.

[9] G. H. Ball and D. J. Hall, "ISODATA, A novel Method of Data Analysis and Pattern Classification," in *Proceedings of the International Communication Conference*, June 1966.

[10] M. Gervautz and W. Purgathofer, "A Simple Method for Color Quantization: Octree Quantization," *Graphics Gems, A.S. Glassner*, pp. 287–293, 1990.

[11] P. Heckbert, "Color Image Quantization for Frame Buffer Display," *Computer Graphics*, vol. 16, no. 3, pp. 297–303, 1982.

[12] M. Chaumont and W. Puech, "A High Capacity Reversible Watermarking Scheme," in *submission*, 2008.

[13] M. Chaumont and W. Puech, "Attack By Colorization of a Grey-Level Image Hiding its Color Palette," in *IEEE International Conference on Multimedia & Expo, ICME'2008*, Hannover, Germany, June 2008.

[14] A. Said (K. Sayood), *Lossless Compression Handbook*, chapter Arithmetic Coding, Academic Press, 2003.