SURVEY ON METHODS OF PROTECTING COLOR INFORMATION BY HIDING IT

Jyothi Lekshmi S[1], Anil A.R[2]

[1]*Department of Computer Science and Engineering, SBCE, Pattoor (India)*

[2]*Assistant Professor, Department of Computer Science and Engineering, SBCE, Pattoor (India)*

ABSTRACT

Conversion from gray to color is known as colorization, this is the process of adding color to monochrome images such as old black and white photos, scientific illustrations etc. The widespread transmission of images and videos makes it necessary to find methods for securing images. Different methods are present for protecting colour information. This paper describes different approaches used. Some method uses color information hiding within the index image, some other methods hides color information in luminance image. A survey on four methods presented here.

Keywords: Index Image, Color Palette, Quantization, Layer Scanning Algorithm, Dwt.

I INTRODUCTION

The widespread transmission of images and videos often makes it necessary to find methods to secure them. For example, applications like confidential transmission, video surveillance, cultural heritage, military and medical applications need security functionalities. Data security thus becomes an important and integral component of modern day research on multimedia information.

Different methods are present for protecting color information. This paper addresses four of those methods.

In the next chapter first solution is palette based approach where color information is protected by hiding the color palette in index image. In that method the colors of the color palette are sorted to get an index image which is near of the luminance of the original color image. In the same time, consecutive colors in the color palette are close.

The other three methods are substitution based approach. These methods are based on wavelet decomposition and sub band substitution propose solutions to embed the color information in a gray level image. Their areas are image printing and perceptive compression. The second method concerned with color documents prepared digitally, which are to be printed using a black-and-white printer or transmitted using a conventional black-and-white fax machine. Therefore, the first problem is how to convert colors to black and white such that different colors would look different on paper too, even if they have the same luminance component. The second problem is the color-to-gray mapping should be reversible.

Third approach is about color embedding and recovery using a pseudorandomized saturation code based on a wavelet packet transforms to improve the color saturation of the recovered color image and preserve as many details as possible from the original image. Fourth method proposes an accurate reversible algorithm without any distortion conditions such as a print-scan process for converting full color images to gray images while preserving the chroma and spatial resolution of the images.

II METHODS OF PROTECTING COLOR INFORMATION

The first method is palette based method. In this method low quality images are freely available but user need a key to get corresponding color image. Other three methods are substitution based approaches. In these methods the color information is protected by hiding it in corresponding luminance image.

2.1. Securing color information of an image by concealing the color palette

This paper presents a method to provide free Internet access to low quality images to all users and restricted access to the same images of better quality with the purchase of a key. That means a solution to give free access to gray level images whereas the user needs a key to view the corresponding image in color. The main objective of this method is thus to protect the color information of an image by embedding it in its corresponding gray level image.

The overview of the method is presented in Fig.1. To quantize the original color image, the luminance image is used in order to choose the number of colors, $K \in N^*$. After quantization procedure, the color reordering algorithm is applied in order to get a reordered color palette and an index image close to the luminance image. The last step consists of embedding the color palette(message), in the index image(cover).

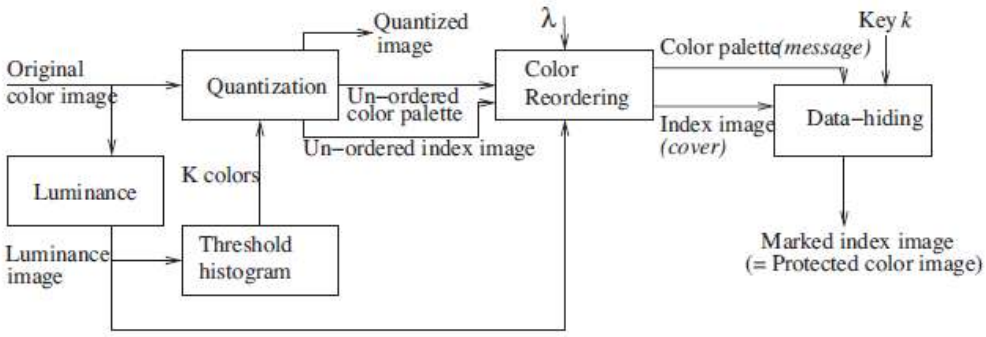

Fig 1: Overview of the method

2.1.1. Color quantization

Quantization means reducing the color number of a color image. The optimal solution, to extract the K colors from an image is obtained by solving Eq. (1):

$$\{P_{i,k}, C(k)\} = arg \min_{P_{i,k}, C(k)} \sum_{i=1}^{N} \sum_{k=1}^{K} P_{i,k}.d^2(I(i), C(k)),$$

$$\text{with } \forall i, \exists! k\prime, P_{i,k\prime} = 1 \text{ and } \forall k \neq k', P_{i,k} = 0, \quad \text{(1)}$$

where I is a color image of dimension N, C(k) is the kth color of the K searched colors, d() is a distance function in the color space, and Pi,k ∈ {0, 1} is the membership value of pixel i to color k. The constraint on Pi,k is that for all i there is a single k such that Pi,k = 1.

A solution to minimize Eq. (1), and then to obtain the K colors, is to use the ISODATA k-mean clustering algorithm. Pi,k is defined in Eq. (2):

$$\forall i, \forall k, P_{i,k} = \begin{cases} 1 & \text{if } k = arg\{\min_{\{k'\}} \; \min_{k' \in \{1,\ldots,K\}} d(I(i), C(k'))\}, \\ 0 & \text{else,} \end{cases}$$

$$\text{with } C(k) = (\textstyle\sum_{i=1}^{N} P_{i,k} \times I(i))/(\sum_{i=1}^{N} P_{i,k}). \quad \text{(2)}$$

In this approach, the number K is significant in comparison to the original number of colors. The problem with classical k-mean algorithm is, number of extracted colo

rs will often be below K, and is known as problem of "death classes". To overcome that problem, initialize the Pi,k values by solving the fuzzy c-mean equation given by Eq. (3):

$$\{P_{i,k}, C(k)\} = arg \min_{P_{i,k}, C(k)} \sum_{i=1}^{N} \sum_{k=1}^{K} P_{i,k}^{m}.d^2(I(i), C(k)), \quad \text{(3)}$$

where m is a fuzzy coefficient (experimentally m is set at 1.6) and Pi,k are real values in the range [0, 1], named fuzzy membership values. Equation (3) is solved by a fuzzy c-mean algorithm. A quantized image is obtained once the quantization with K colors has been carried out. A color palette and its index image are associated with this quantized image. By applying just the quantization algorithm, the content of the index image does not semantically correspond to the luminance content of the original image. The color palette order should be changed to obtain an index image where its content are semantically close to the luminance content. Consequently the associated index image will change too.

2.1.2. Layer scanning algorithm

After color quantization, the K color image can be represented by an index image (based on Pi,k values) and a color palette (based on C(k) values). The index image is denoted index and is defined such that

$$\forall i \in \{1, \ldots, N\}, \text{index}(i) = arg \max_{k \in \{1,\ldots,K\}} P_{i,k}. \quad \text{(4)}$$

The color palette is denoted palette and ∀ k ∈ {1, . . ., K},

$$Pallette(k) = C(k)$$

The goal is then to find a solution by taking two constraints into account. The first constraint is to get an index image where each gray level is close to the luminance of the original color image. The second constraint is that two consecutive colors should be close in the color palette. The problem is to find a permutation function that simultaneously permutes the values of the index image and those of the color palette. The best permutation function is found by solving Eq. (5):

$$\Phi = \underset{\Phi}{\operatorname{argmin}} \left[\sum_{i=1}^{N} E_i^{ind} + \lambda \sum_{k=1}^{K-1} E_k^{palette} \right], \quad (5)$$

where E_I^{ind} (equation(6)) is the energy related to the index image and $E_k^{palette}$ (equation(7)) is the energy related to the color palette:

$$E_i^{\text{ind}} = d^2(Y(i), \Phi(\text{index}(i))),$$
$$E_k^{\text{palette}} = d^2(\text{palette}(\Phi^{-1}(k)), \quad \text{palette}(\Phi^{-1}(k+1)))],$$

where Y is the luminance of the original color image, and $\lambda \in R_+^*$ is a parameter controlling the weight assigned to the two energy terms. The Φ permutation function is a bijective function in N defined by $\Phi : \{1, \ldots, K\} \rightarrow \{1, \ldots, K\}$. The minimization of Eq. (7) is not feasible by using a derivative approach since Φ(·) and palette(·) are discrete functions. A metaheuristic approach could be used, such as evolutionist algorithms, for minimizing Eq. (5), but we prefer a less CPU consuming solution and a less memory costly solution. Eq. (5) is thus solved using an heuristic algorithm: the layer scanning algorithm. The aim of this algorithm is to find a reordering of K colors such that consecutive colors are close and the colors are reordered from the darkest to the lightest. This reordering defines, for each kth color, a k' position which gives the Φ function such that Φ(k) = k'.

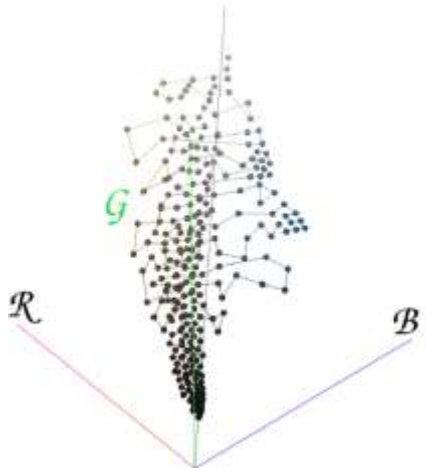

Fig 2: A view of the path built with the layer scanning algorithm in the RGB space.

To reorder the K colors, the algorithm scans the color space to build the reordered suite of colors. This scanning is obtained by jumping from color to color in the color space, and then choosing the closest color to the current one. The first color of this suite is chosen as the darkest one among the K colors. An additional constraint to this scanning

is that the search is restricted to colors which are not too different in terms of luminance. This means that the scanning in the color space is limited to a layer defined on luminance information. This layer scanning algorithm could then be seen as a kind of "3D spiral scan" in the color space.

With the application of this layer scanning algorithm, we obtain a reordered color palette and its index image, which is semantically understandable. Note that the application of this algorithm does not change the informational content. Indeed, this new color palette and the associated index image allow for the same color image to be built before and after processing the layer scanning algorithm. The energy shown in Eq. (6) is related to the index image as a function of the layer size. This energy is minimized when the layer size is small. For example, if the layer size is equal to 1 then the index image is built mainly as a function of the luminance of the palette colors. For the energy related to the color palette, Eq. (7), the variation as a function of the layer size differs from the previous one. In Eq. (5), parameter λ could give, as a function of its value, more importance to the color palette continuity or to the index image.

This layer running algorithm has an implicit hidden parameter which is the layer size used during the color running in the color space. Since our goal is to minimize Eq. (5), a satisfactory way to automatically set this parameter is to test all possible values for this layer size parameter and to keep the layer size value minimizing the equation. Knowing that the possible values of the layer size parameter belong to the range {1, . . ., K } and that it is very fast to make just one run in the color space, this gives an elegant and rapid solution to approximate Eq. (5).

Practically, λ depends on the parameter, $\alpha \in R^{*}_{+}$ given in Eq. (8):

$$\lambda = \alpha * \frac{N}{K - 1} \quad (8)$$

where N is the image size in pixels. With α set at 1, the same weight is given to the two constraints. With $\alpha \in [0,1]$, more importance is given to the luminance constraint, and inversely with ˛ greater than 1, more importance is given to the continuity of the color palette. Note that alpha is set by the user.

2.1.3. Color protection of an image by data-hiding

Embed data of the color palette into the index image. Spatial domain methods embed the information directly into the pixels of the original image. The main steps of this data hiding algorithm are the partitioning of the image into blocks of homogeneous size, the selection of one pixel in each block and the substitution of the gray level of each selected pixel by one of its two neighbors of the color palette. This is known as index substitution.

The objective of the data hiding step is thus to embed a message M made up of m bits b_j(M = b1b2 . . . bm). The embedding factor, in bit/pixel, is given by Eq. (10):

$E_f = m/N$ …….. (10)

The original index image is then partitioned into blocks of size $1/E_f$ pixels. Each block is used to hide only one bit b_j of the message. This partition procedure guarantees that the message is spread homogeneously over the whole image. In order to hide the data of the color palette within the image, m = 3 ×K×8 bits must be embedded in the

index image. When K = 256 colors (which is the maximal value), the number of bits to embed is m = 6144 bits and, for an image of 512 × 512 pixels, the size of the blocks is 42 pixels and the embedding factor E_f = 0.0234 bit/pixel. To get an embedded pixel, $index_w(i)$, the selected pixel index(i) is then substituted by one of its two neighbors according to the message bit bj in order to preserve the best quality of the reconstructed image from the data-hidden image, as formalized by Eq. (11):

$index_w(i)$ =

$$\begin{cases} index(i) \; if \; bj = index(i) \bmod 2 \\ argmin \; (k \in [index(i)1, index(i)1]n \neq \{1 \ldots . k\})(palette(index(i) - palette(k))2 \; otherwise \end{cases}$$

……(11)

Thus, the index value index(i) is modified by +1 or −1 gray level when bj index(i) mod 2. The best choice for this modification is then to choose the closest color between palette(index(i) + 1) and palette(index(i) − 1) in order to minimize the distance to the color palette(index(i)). This way of embedding the color palette ensures that each embedding pixel can be modified only by one gray level and at the same time the reconstructed color pixel from the embedding image is very close to the original color value.

2.2. Color to Gray and Back: Color Embedding Into Textured Gray Images

Here approach is to map colors to textures. That is regions of different colors with similar luminance will look different after mapping because they would have different textures .Instead of having a dictionary (or palette) of textures and colors, it produces a continuum of textures that naturally switch between patterns without causing visual artifacts. For that, make use of the discrete wavelet transform (DWT), which decomposes an image into several subbands, each representing different spatial frequency contents. This wavelet-based mapping method works as follows.

2.2.1. Color Embedding

i. Convert image from RGB into YCbCr (or CIELab).

ii. Use a two-level DWT on Y , so that is divided into seven subbands:
Y→(S1,Sh1,Sv1,Sd1,Sh2,Sv2,Sd2)

iii. Reduce Cb and Cr by ½ , construct Cb+, Cb-, Cr+, Cr-, and reduce Cb- further to ¼ of its original size.

iv. Replace subbands
Sd1←Cb-; Sh2←Cr+; Sv2←Cb+; Sd2←Cr-

v. Take inverse DWT to obtain the textured gray image, i.e.,
(S1,Sh,Sv,Cb-,Cr+,Cb+,Cr-)→Y'

vi. Image Y' is the resulting gray image and may be printed, which often includes scaling and halftoning.

2.2.2. Color Recovery

i. Read or scan the gray textured image.
ii. Determine image dimensions.
iii. If necessary, identify corners and carry an affine transform to de-warp the gray image.
iv. Reduce image to the correct resolution.
v. Use a DWT to convert the gray image into subbands
Y'→(S1,Sh1,Sv1,Sd1,Sh2,Sv2,Sd2)
vi. Interpolate Sd1, doubling its resolution.
vii. Make Cb=|Sv2|-|Sd1|, and Cr=|Sh2|-|Sd2| .
viii. Interpolate Cb and Cr, doubling their resolutions.
ix. Remove the embedded subbands, i.e., set Sd1=Sh2=Sv2=Sd2=0, and take the inverse DWT transform to Y find as
(S1,Sh1,Sv1,0,0,0,0)→Y
x. Convert the Y, Cb, Cr planes back to RGB.

The reason to create positive and negative-valued chrominance planes is to avoid completely the color inversion problem depicted in Fig. 6. If a subband is supposed to have only positive values and we obtained negative ones, then it is a sign of texture inversion. Hence, one can take the absolute value of the subbands and recombine them into Cb and Cr as Cb=|Cb+|-|Cb-| and Cr=|Cr+|-|Cr-|.

2.3. Color Embedding and Recovery Using Wavelet Packet Transform with Pseudo randomized Saturation Code

This article proposes color embedding and recovery using a pseudorandomized saturation code based on a wavelet packet transform to improve the color saturation of the recovered color image and preserve as many details as possible from the original image. In the color-to-gray process, the Y image is divided into 16 subbands using a two-level wavelet packet transform. To minimize the loss of detail, the CbCr color components are then embedded into the two subbands with the minimum amount of energy in the Y image, where the selected combination of subbands is determined by investigating various combinations of eight subbands. Furthermore, to compensate the color saturation, the Cb and Cr components are scaled using the maximum and minimum values of the CbCr components from the original image. These values are embedded into the diagonaldiagonal (DD) subband and transformed into a pseudorandom code while considering the visibility in the resulting gray image. In other words, the pseudorandom code includes the maximum and minimum values of the CbCr components in original image and is expressed using numbers of white pixels. However, to reduce the degradation of details when using the pseudorandom code in the wavelet packet transform, the number of pixels is down sampled in the diagonal-diagonal subband. Finally, the ratio of the original CbCr values to the extracted CbCr values is applied to enhance the saturation of the recovered color image.

The method used to recover the colors from the new gray image with texture is the reverse process of the above color-to-gray algorithm.

2.3.1. Subband Selection for Embedding CbCr

Figure 3 shows the subband locations selected for embedding the CbCr components.

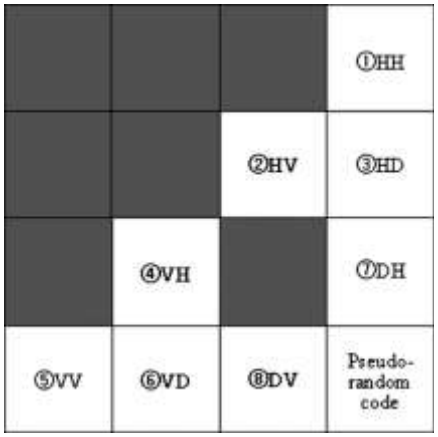

Fig 3: Subband locations for embedding the CbCr components.

Here, eight candidate subbands were selected: the horizontal-horizontal (HH), horizontal-vertical (HV), horizontal-diagonal (HD), verticalhorizontal (VH), vertical-vertical (VV), vertical-diagonal (VD), diagonal-horizontal (DH), and diagonal-vertical (DV) subbands. The seven subbands in the dark region in Fig. 3 were excluded, as they included relatively much more information on the original image than the other subbands. The DD subband was also excluded, as it was used to embed the pseudorandom code to compensate for the color saturation. Thus, the eight candidate subbands yielded 28 possible combinations of two subbands for embedding the CbCr components. The subbands used to embed the Cb and Cr components are selected based on their color difference in CIELAB color space and peak signal-to-noise (PSNR) values. Thus, the color differences in CIELAB color space and PSNR values were calculated as follows:

$$\Delta E^*_{ab} = (\Delta L^{*2} + \Delta a^{*2} + \Delta b^{*2})\wedge(\frac{1}{2})$$

where

$$\Delta L^* = L2 - L1, \quad \Delta a^* = a2 - a1, \quad \Delta b^* = b2 - b1$$

$$PSNR_k = 20\ log_{10}\left(\frac{255}{\sqrt{MSE_k}}\right), k = R, G, B$$

Where

$$MSE_k = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}||O_k(i,j) - R_k(i,j)||^2$$

The combination of two subbands that resulted in the best performance as regard to the color difference and PSNR value depended on the image. The best color difference and PSNR value were obtained when using combinations among the HV, VH, DH, and DV subbands possibly because they included the minimum information on the original

image. Therefore, combinations of two subbands among the HV, VH, DH, and DV subbands produced the best recovered color image, the CbCr components were always embedded into the HV and VH subbands. To reduce the interpolation errors in the gray-to-color process, one-quarter of the CbCr images can be used instead of onesixteenth of the individual CbCr images. The quarter-sized CbCr images are divided into four regions of the same size instead of down sampling to one-sixteenth of the size CbCr images. Each separate region of the Cb image is then embedded into the DH, HH, HV, and HD subbands. Meanwhile, each separate region of the Cr image is embedded into the DV, VH, VV, and VD subbands, respectively. Next, an inverse wavelet transform is applied to obtain a textured gray image.

2.3.2. Embedding a Pseudorandom Code for Compensation of Saturation

When a textured gray image is printed and scanned, the pixel values (eight bits: 0–255) of the textured gray image are invariably changed by the inherent characteristics of the printer and scanner. This effect leads to a loss of color saturation in the recovered color image as the values of the embedded CbCr components are also changed in the printing and scanning process. Thus, to enhance the color saturation of the recovered color image, the CbCr components were scaled using the maximum and minimum values of the CbCr components of the original image. When using the ratio of the original CbCr values to the extracted CbCr values, the color saturation of the recovered color image can be enhanced. Therefore, this information is transformed into a
pseudorandom code, considering the visibility of the textured patterns on the new gray image, and then embedded into the DD subband.

2.3.3. *Generation of Pseudorandom Code*

The pseudorandom code is expressed by the number of white pixels according to a given code book, and the location of each pixel is chosen pseudorandomly. First, an image is generated that is the same size as each wavelet packet subband. This image is then divided into 16 regions with 4×4 blocks of the same size, and the subband pixel values are set to zero. Next, the pseudorandom code is generated using the maximum and minimum values of the CbCr components. Although the range of CbCr values is generally −128–+127, the range may be changed to −64–+63 to allow more effective representation of the CbCr information by the pseudorandom code. Next, the tens digit is separated into two parts, while the units digit is divided by a factor of 2. As a result, the first row of the pseudorandom code represents the sign information, the second and third rows represent the tens digit information, and the last row represents the units digit information. Meanwhile, the first and second columns of the pseudorandom code are the maximum and minimum Cb values, respectively, while the third and last columns are the maximum and minimum Cr values, respectively.
Registration and geometric distortions are problems addressed in this method. The next method solves this.

2.4. Accurate reversible color-to-gray mapping algorithm without distortion conditions

This paper propose a new color-to-gray mapping algorithm and a color recovery method that preserves the chroma and the spatial resolution of the original image without distortion conditions.

2.4.1. Color-to-gray conversion

This method works as follows:

step E1: Convert images from RGB to YCbCr.

Step E2: One-level discrete wavelet transform on the luminance Y.

Y →(Sl, Sh1, Sv1, Sd1)

Step E3: Reduce Cb and Cr by 1/16, construct Cb+, Cb_, Cr+, and Cr_ where Cb is resolved into two components depending on whether its sign is positive or negative by the following equations:

$$Cb+ = \begin{cases} Cb, & Cb > 0 \\ 0, & Cb \leq 0 \end{cases}$$

$$Cb- = \begin{cases} Cb, & Cb < 0 \\ 0, & Cb \geq 0 \end{cases}$$

The same arrangement is made for Cr.

Step E4: Multiply b(0 < b 6 1) by the chrominance components Cb+, Cb_, Cr+, and Cr_. Hereafter, this process is referred to as a simple scaling so called b-transform.

Step E5: Each subband Sh1, Sv1 and Sd1 is composed of blocks that consist of 2 × 2[pixels]. In the case of the horizontal subband Sh1 and the vertical subband Sv1, one pixel in the block is replaced by the chrominance component of the corresponding position to obtain Sh1' and Sv1', respectively. In the case of the diagonal subband Sd1, two pixels in each 2 ×2 block are replaced by the corresponding chrominance components to obtain Sd1'.

Step E6: Replace subbands

Sh1← Sh1'; Sv1← Sv1'; Sd1 ←Sd1'

Step E7: Take the inverse discrete wavelet transform to obtain the textured gray image, i.e.,

(Sl, Sh1', Sv1', Sd1')→Y

2.4.2. Recovery step

Step R1: Read the gray textured image.

Step R2: Use a discrete wavelet transform to convert the gray image into subbands:

Y'→(Sl, Sh1', Sv1', Sd1')

Step R3: Obtain Cb-',Cr+',Cb+' and Cr-' from subbands Sh1', Sv1' and Sd1'.

Step R4: Estimate the high-frequency components of the color-embedded pixels in subbands by interpolation. Obtain Sh1" by interpolating the color-embedded pixels in Sh1' in the vertical direction. Obtain Sv1" by interpolating color-embedded pixels in Sv1' for horizontal direction. Make Sd1" by interpolating the color-embedded pixels Sd1' as the mean value of two high-frequency components in a 2 ×2 block.

Step R5: Use an inverse discrete wavelet transform to recover Y"

(Sl, Sh1", Sv1", Sd1"→Y"

Step R6: Multiply 1/b by Cb-'; Cr+'; Cb+', and Cr-'.

Step R7: Obtain Cb" from Cb+' and Cb-', and Cr" from Cr+' and Cr-'. Then Interpolate Cb" and Cr" linearly,by quadrupling their resolutions.

Step R8: Convert the Y" , Cb" , and Cr" planes back to RGB.

This method is constructed on the basis of wavelet transforms and the chrominance information is embedded into the wavelet subbands. The power in the high-frequency subbands Sh1, Sv1 and Sd1 is smaller than that in the low-frequency components. In this algorithm, the chrominance components Cb+, Cb_, Cr+ and Cr_ are spatially distributed and embedded in the above one-level wavelet subbands Sh1, Sv1 and Sd1 effectively. Two chrominance components are embedded in subband Sd1, because power of Sd1 is generally the smallest among the high-frequency subbands.

III CONCLUSION

In this chapter, we proposed different methods for protecting color information. One palette based approach and three substitution based approaches are discussed. Protecting the color information providing a degraded gray-level image, and a rebuilt color image of good quality is possible. Improvements are possible by mixing the different approaches or with prior knowledge from embedding and extracting side.

REFERENCES

[1]. M. Chaumont and W. Puech, "A Grey-Level Image Embedding its Color Palette," Proc. ICIP, vol.I, pp.389-392, 2007.

[2]. M. Chaumont and W. Puech, "A Fast and Efficient Method to Protect Color Images," Proc. IS&T/SPIE Symp. Electronic Imaging, VCIP, 6508, 65081T, 2007.

[3]. M. Chaumont and W. Puech, "Protecting the Color Information by Hiding It," in Recent Advances in Signal Processing, ISBN 978-953- 7619-41-1, 2009.

[4]. M. Chaumonta, W. Puechb and C. Lahanier, "Securing Color Information of an Image by Concealing the Color Palette," Journal of Systems and Software, vol. 86, no. 3, pp. 809–825, 2013.

[5]. R. L. de Queiroz and K. Braun, "Color to Gray and Back: Color Embedding Into Textured Gray Images," IEEE Trans. Image Process. Vol. 15, no. 6, pp. 1464-1470, 2006.

[6]. K.-W. Ko, O.-S. Kwon, C.-H. Son and Y.-H. Ha, "Color Embedding and Recovery Based on Wavelet Packet Transform, Journal of Imaging Sc. and Tech.," vol. 52, no. 1, pp. 010501-{1 -10}, 2007.

[7]. T. Horiuchi, F. Nohara and S. Tominaga, "Accurate Reversible Colorto- Gray Mapping Algorithm without Distortion Conditions," Pattern Recognition Letters vol.31,pp. 2405–2414, 2010.