2005 IEEE International
Symposium on Signal Processing
and Information Technology

Deliberate Distortion of Color Image and Video Resources for Copyright Protection

Arash Abadpour [1] and Shohreh Kasaei [2]
[1] Mathematics Science Department, Sharif Univ. of Tech., Tehran, Iran, email: abadpour@math.sharif.edu
[2] Computer Engineering Department, Sharif Univ. of Tech., Tehran, Iran, email: skasaei@sharif.edu

Abstract— **The easiness of data–flow in digital media, seriously claims the ownership of intellectual material. To solve this problem, researchers have worked on different watermarking methods to embed ownership data into original signals. Unfortunately, none of the available techniques have been able to assure an acceptable level of security. Though, recently some senior members of the image processing community are expressing essential doubt about the appropriateness of watermarking for data protection and declare it more as a nonsecure tool for data embedding. In this paper, we propose a fast method for copy protection of color visual objects. The proposed method, deliberately embeds fake edges and color alterations into a given a color image in the way that it carries the content of the original image. Using this damaged image prevents the unauthorized users to pirate the demo version of images and video placed at homepages to help consumers select the images they wish to purchase. The main contribution of the proposed method is that there is a small key for the distorted image that reverts all degradations in a lossless manner. To the knowledge of the authors, this is the first time this kind of copy protection is addressed. While there are literally infinite ways to encrypt a given image, it is proved both experimentally and mathematically that there is no practical chance to crack the code.**

I. Introduction

Although, establishment of digital media has been beneficial for enhancing data–flow in the human community, it has resulted in some major drawbacks. In this way, the easiness of storage and doubling of data in digital frameworks, is essentially claiming the copyright integrity of intellectual material. This is the main motivation for researchers to investigate efficient ways to save the ownership of multimedia streams. [1], [2].

One of the main branches of multimedia, is the pictorial data in form of still and moving pictures. Recently, in April 2005, the giants of motion pictures started a serious movements towards banning peer-to–peer sharing of video on the internet. While trials like this have been at most partially successful in providing a safe atmosphere for copy protection, researchers are getting more and more aware to give an effective solution. For example, in watermarking, a legal data is added to the original data to help identifying the original owner. Historically, during last decades different approaches for image (*e.g.*, see [3], [4], [5], [6]) and video (*e.g.*, see [7], [8], [9], [10], [11]) watermarking have been developed. Though, none of them is claimed to be ultimately resistent to all kinds of attacks. Note that the meaningfulness of the "attack–resistance", reported for many of the available approaches, should be carefully regarded (for a full analysis of attack–resistance see [12], [13], [14] and *StirMark* [15]). In fact, development of cheating methods and the range of expert people taking part on them, makes it undoubtable that for any watermarking method, there is generally affordable methods of attack to produce non–watermarked images or even embed another watermark [16].

Recently, the integrity of the watermarking methodology and its properness for security is essentially criticized [17]. In the afterwards comments on *Herley*'s controversial note "*Why Watermarking is Nonsense*" [17], researchers emphasized that this young field of signal processing is "*oversold*" and that no method has yet been able to claim "*the ability to protect from all possible future attacks*" [18], [19]. Researchers emphasize on nonsecurity–oriented application of watermarking and count on the new methods to come to help in the security field [19]. By the way, *M. Barni* makes the point clear "*Why should we hide the information within the data, when we could more easily use headers or other means to reach the same goal?*" [20]. In our opinion this forum is still open. See [21] for a survey of security scenarios in music watermarking and their failure.

After all, it seems that the classic methods of data protection are still the best choices. For example, see [22] for a set of guidelines for ultimate image copy protection from a graphics specialist in NASA, containing quotes like "*never expose an image in its large size*", and suggesting to use "*visual watermarks*" and "*programming shield*".

In a much different approach, a few researchers work on direct copy detection [23], but it is not yet an stable practical tool, too.

In this paper, we address the problem of protecting the ownership of a visual object which should exposed to the public. As an application, assume the agencies selling images in their webpages (*e.g.*, *webshots.com*). the images should be presented to possible consumers, but the viewers should not be able to save the images for their own and then probably use them in professional publication. Although, there are some programming shields for this purpose, we believe they all will be soon cracked, as similar guards did. The classic solution to this problem is to put the images in a restricted manner, for examples, cropped, badly compressed, or down–sampled. In this way, for selling image I, the agency should reserve room for two images. While the original image, I, should be saved in a high–security zone, system should produce and expose $\tilde{I}$, the restricted version of I, to the public. In this strategy, a overall redundancy of $\|\tilde{I}\|/\|I\|$ is forced to the system. where,

$\|I\|$ is the volume of I. Also, there is an ever present threat that the dataset of original images, which is extremely large, may be cracked.

The proposed solution is a lossless method of degrading the color content of an image or video. The degradation is performed using a random key generated by the encrypter. The resulting image, conveys the information of the original image, but contains fake edges and unrealistic altered colors. An important achievement of the method is it's low computational cost and the small size of the key, while its reproduction is literally impossible.

The rest of this paper is organized as follows: Section II introduces the proposed copy protection method and Section III holds the experimental results. Finally, Section IV concludes the paper.

II. Proposed Method

In this section, first some mathematical preliminaries are reviewed and then the proposed method is introduced. Section II-A states the way to show a right—rotating orthonormal matrix as three angles and discusses how to reconstruct it back from those angles. Section II-B briefly discusses the method to produce a uniform quantized random variable which may be losslessly saved in a few–bits representation. Then, Section II-C incorporates the above methods in the proposed method for color image copy protection. Finally, Section II-D discusses the quality measures used in this paper.

A. Polarization and Depolarization

There is a manipulated form of the common *Euler* angles that relates any right–rotating orthonormal matrix (such as V_{ij}) with three angles, in a one–to–one revertible transformation [24].

A right–rotating orthonormal matrix is the orthonormal matrix satisfy, $(\vec{v}_1 \times \vec{v}_2) \cdot \vec{v}_3 > 0$, where, $\times$ and $\cdot$ represent outer and inner products, respectively, and v_i is the i–th column of V. In this way, for the right–rotating orthonormal matrix V we write $V \sim (\theta, \phi, \psi)$ when,

$$\theta = \angle\big(\vec{v}_1^{xy}, [1,0]^T\big), \tag{1}$$

$$\phi = \angle\Big(\big(R_\theta^{xy}\vec{v}_1\big)^{xz}, [1,0]^T\Big), \tag{2}$$

$$\psi = \angle\Big(\big(R_\phi^{xz}R_\theta^{xy}\vec{v}_2\big)^{yz}, [1,0]^T\Big). \tag{3}$$

Here, $\angle(\vec{v}, \vec{u})$ denotes the angle between two vectors $\vec{v}, \vec{u} \in R^2$, computed as:

$$\angle(\vec{v}, \vec{u}) = sgn((\vec{v} \times \vec{u}) \cdot \vec{j}) \cos^{-1} \frac{\vec{v} \cdot \vec{u}}{\|\vec{v}\|\|\vec{u}\|} \tag{4}$$

where $sgn(x)$ is the *signum* function, defined as:

$$sgn(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}. \tag{5}$$

In (1), (2), and (3), $\vec{v}_p$ denotes the project of the vector v on the plane p. Also, R_α^p is the 3×3 matrix of α radians counter–clock–wise rotation in the p plane:

$$R_\theta^{xy} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{6}$$

$$R_\phi^{xz} = \begin{pmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{pmatrix}, \tag{7}$$

$$R_\psi^{yz} = \begin{pmatrix} 0 & \cos\psi & -\sin\psi \\ 1 & 0 & 0 \\ 0 & \sin\psi & \cos\psi \end{pmatrix}. \tag{8}$$

One can prove that

$$R_\psi^{yz} R_\phi^{xz} R_\theta^{xy} V = I. \tag{9}$$

Hence, to produce V out of the triple (θ, ϕ, ψ) one may use the below equation,

$$V = R_{-\theta}^{xy} R_{-\phi}^{xz} R_{-\psi}^{yz}. \tag{10}$$

Note that $(R_\alpha^p)^{-1} = R_{-\alpha}^p$.

While equations (1), (2), and (3) compute the three angles θ, ϕ, and ψ out of the V, equation (10) reconstructs V from θ, ϕ, and ψ. These methods are called *polarization* and *depolarization* of a right–rotating orthonormal matrix, respectively [24].

B. Uniform Quantized Random Variable

Assume that the random variable r is given which is uniform in the range $(0, 1]$. We compute the new random variable s as,

$$s = A\left(2\frac{\lceil 2^q r\rceil}{2^q} - 1\right). \tag{11}$$

Now, s is a uniform random variable getting the 2^q values of,

$$s \in \Big\{2^{1-q}kA - A\Big|k = 1 \cdots 2^q\Big\}. \tag{12}$$

Here, q is the number of quantization steps of s and A is its maximum amplitude. Now, assume computing,

$$k = 2^{q-1}\Big(1 + \frac{s}{A}\Big). \tag{13}$$

Simple manipulations show that k is a q–bit unsigned integer.

Thus, having a random generator giving a uniform random variable in the range $(0, 1]$, we propose to use (11) to have a new uniform random variable in the range $(-A, A]$. The importance of the new random variable is that its values are represented in q–bits in a lossless manner. In contrast, for near lossless storage of the original random variable, one should spare 64–bit of memory for storing a double precision floating point number.

C. Block–based Copy Protection

Assume that the $H \times W$ color image I is given. I is first split into non–overlapping rectangular areas of size $h \times w$, where h and w are given by the user. Here, we put some restrictions for the choice of h and w. The two common compression techniques of JPEG and JPEG2000 work on 8×8 and 32×32 blocks of the given images, respectively. Thus, we urge w and h to be selected in the way that $m|w, h$, where $x|y$ means x divides y and m is the block–size of the utilized compression method (apparently, $m = 8$ for JPEG and $m = 32$ for JPEG2000). We will return to the reason for this restriction. As the images in the CIP dataset [25] are 512×512 images, the user is asked for an integer N in the range of $[0, 9 - \log_2 m]$. Then h and w are selected as $H2^{-N}$ and $W2^{-N}$, respectively. In this, way the image is split into 2^{2N} rectangular regions.

Let's return to the problem in hand. Call the the expectation vector of the block I_{ij} as $\vec{\eta}_{ij}$. We propose to use three values of a uniform quantized random variable r as θ_{ij}, ϕ_{ij}, and ψ_{ij} (see Section II-B). These values are produced using the value of A, which is given by the user in the range $[0, 2\pi]$ and the value of $q \in [0, 8]$. We will discuss these parameters and their rule in the results. Now, assume that $V_{ij} \sim (\theta_{ij}, \phi_{ij}, \psi_{ij})$ is computed as discussed in Section II-A. For each color vector $\vec{c}$ in I_{ij}, we compute the new color vector $\vec{d}$ as,

$$\vec{d} = V_{ij}(\vec{c} - \vec{\eta}_{ij}) + \vec{\eta}_{ij}. \tag{14}$$

This process is performed for all blocks and all pixels within them. The result is the image I_e, the image equivalent to I in its content, but containing deliberately embedded fake edges and color distortions. the parameters for this process are N (number of blocks), A (amplitude), and q (quantization levels). The result of the process is the encrypted image I_e, plus three $2^N \times 2^N$ arrays of q–bit values for storing θ_{ij}, ϕ_{ij}, and ψ_{ij}. As I_e may contain values out of the $[0, 255]$ range, which are cut when I_e is to be saved in a standard format like JPEG, we perform a linear scaling on it. The parameters of the scaling (a bias and a scale) are also saved.

To decrypt an image it is enough to revert the scaling, and then to use the values of θ_{ij}, ϕ_{ij}, and ψ_{ij} to reproduce V_{ij}. Then using,

$$\vec{c} = V_{ij}^T(\vec{d} - \vec{\eta}_{ij}) + \vec{\eta}_{ij}, \tag{15}$$

the original image is reconstructed. Note that the expectation vector of the ij–th block in I and I_e are identical. Hence, η_{ij} should not be sent to the decrypter. Also note that, unless for numerical errors and truncations, nothing is lost in this process. We call the image produced by the decrypter as I_d. Also, the intervals elapsed on performing encryption and decryption are called t_e and t_d, respectively. Figure 1 shows a flow–chart of the proposed method and Section III discusses the computational complexity of the method and the meaning of the parameters. Also, sample utilizations of the proposed method are given there.

When an image I is encrypted into the image I_e, it should be exposed to the consumers. Hence, a standard image format should convey the contents of I_e. As the dynamic range of I_e is changed to $[0, 255]$, saving I_e as a BMP file only results in truncations, which is an acceptable distortion (gives a PSNR of $54dB$). The situation is more crucial when I_e is going to be saved is a more professional format like JPEG or JPEG2000. The proposed method well tolerates this situation. In fact I_e contains consistent blocks of size $h \times w$, and the inconsistencies only occur at the edges of these blocks. In accordance, JPEG and JPEG2000 divide an image into non–overlapping blocks and perform corresponding transform on these blocks, separately. Hence, any non–stationarity caused within a block will result in a low–compression rate or low–quality event in the result of compression. By retaining the $m|h, w$ condition, the process makes sure that no in–stationarity occurs inside a block. Hence, I_e may be fed to a JPEG–based compression module with no aftermath on the quality of I_d, except for the losses produced by the compression itself. We emphasize that neglecting the $m|w, h$ condition results in sever losses. One important point here is that, in the JPEG standard there is the possibility to down–sample the chrominance planes by half or one–fourth. Using this option will result in a fake grid in I_d and should be prohibited. Section III discusses the adaptivity of the proposed method with compression methods with illustrative examples.

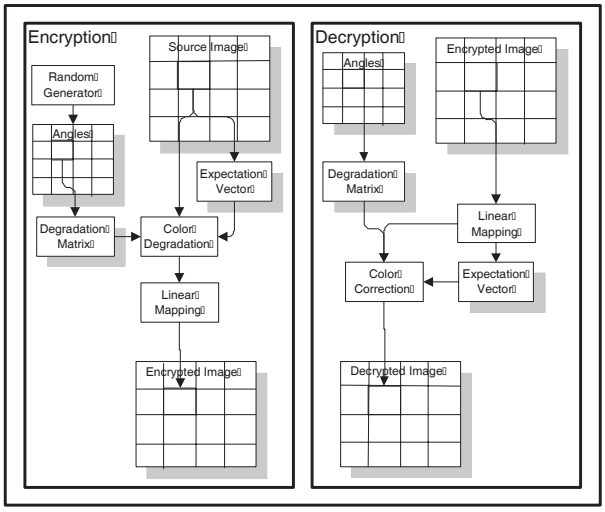

Fig. 1. Flowchart of the proposed method.

D. Quality Measures

The classic measure of quality is the *peak signal to noise ratio* (PSNR), computed as,

$$PSNR = 20\log_{10}\frac{255}{\sqrt{E\{(I - \tilde{I})^2\}}}, \tag{16}$$

where, I is the original signal and $\tilde{I}$ is the degraded one. Here, it is assumed that the dynamic range of both I and $\tilde{I}$ are $[0, 255]$. Added to this measure, we use another measure,

called maximum deviation, defined as,

$$Max.\ Dev. = 20\log_{10}\frac{255}{\sqrt{\max\left\{(I-\tilde{I})^2\right\}}}. \qquad (17)$$

While maximum deviation is always less than (rarely equal to) PSNR, it reveals the worst degradation of the input signal. In this way, there is a possibility to have a signal with high PSNR, while it contains very small portions which are massively degraded. For these signals the fall of maximum deviation will be a helpful indicator.

III. Experimental Results

The experimental results are carried out in a 2046MB PIV processor using MATLAB 6.5 and image processing toolbox 3.2.

The whole information used for encrypting I into I_e includes N, A, q, scale, bias, and $3q2^{2N}$ bits for the distortion angles. While each of A and q are represented using a nibble, writing A as $2^{-a}\times 2\pi$, $a\in\{0,\cdots,7\}$ also lets A to be stored in a nibble. Hence, the total size of the encryption key equals $3q2^{2N}+32$ bits (2 bytes are used for string the minimum and maximum of I_e to reconstruct the scale and bias). As for a known q there are q choices for each angel, there are $2^{3q2^{2N}}$ ways to encrypt a single image. In another view, there is a $2^{-3q2^{2N}}$ chance to crack the key of an image using a single trial. For a nominal value of $q=3$ and $N=3$, the total size of the key equals 76 bytes with 2.4×10^{173} possible solutions and 4.2×10^{-172} chance for cracking the code by a single trial. While, the chance to crack the code for a single block is 2^{-3q} (In this nominal case, 0.002), we emphasize that independent decryption of different blocks results in visible grids in the image. Hence, we argue that it is definitely impossible to crack the code by trial–and–error, even in these small choices for q and N.

Now, lets compte the computational complexity of the proposed method. In the encryption stage, each block is first separately processed. In this way, the processor should spent $3WH2^{-2N}$ flops on processing the expectation vector of a block and then 3 flops to compute the three angles. After spending 87 flops for computing V_{ij}, $15WH2^{-2N}$ flops are needed to perform the operation in (14). Adding up $3WH$ flops for computing the scale and bias and applying them, the total flops needed to perform the proposed encryption process on an image equals $21WH+105\times2^{2N}$. For nominal values of $W=H=512$ and $N=5$, 105×2^{2N} is less than two percents of $21WH$, hence, the total computation cost of the proposed encryption process equals $21WH$ flops, independent of N. The load for decrypting an encrypted image is equal to the above figure, except for $2WH$ flops spent on finding the bias and scale. Hence, decryption process elapses $19WH$ flops. In this way, the computational complexity of both process are linearly relating to the image size. To have an idea about the elapsed time of the proposed process on a DSP processor, we assume implementation of the method on a one Giga flop per second processor like $DIOPSIS^{TM}740$ by *Atmel* [26] or $TMS320C6713$ by *Texas Instruments* [27]. Encrypting a 512×512 image in these platforms elapses less than $6ms$ ($5ms$ for decryption). Hence, real–time video encryption using the proposed method only elapses 15% of the processor's time. Note that here the interesting features of this processors for pipelining and SIMD operation are ignored.

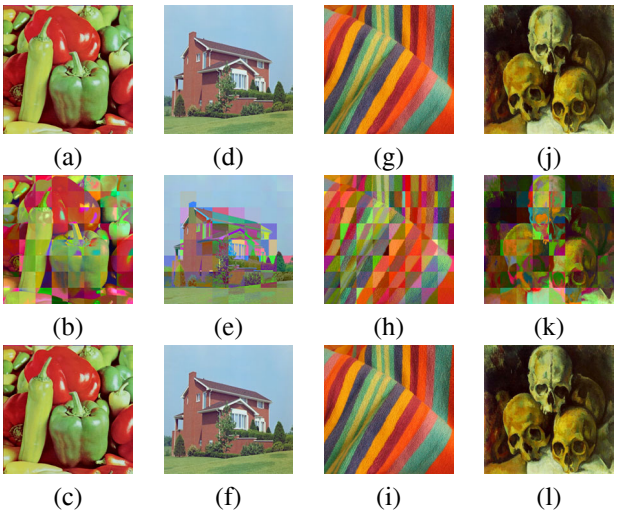

Fig. 2. Sample results of the proposed encryption/decryption method. (a), (d), (g), and (j) Original images. (b), (e), (h), and (k) Encrypted images. (c), (f), (i), and (l) Decrypted images. (a) *Peppers*, courtesy of Signal and Image Processing Institute at the University of Southern California. (d) *House*, courtesy of The Ohio State University. (g) *Skulls*, adopted from *http://www.artchive.com* .

Figure 2 shows a few images and their corresponding encrypted versions. The parameters in this experiment are set as $N=3$, $A=\frac{\pi}{2}$, and $q=4$. While the key length equals 100 bytes (0.0127% of the image size), the chance for cracking the code by a single trial is less than 10^{-231}. In this case, t_e is less than $400ms$ and t_d is about $320ms$. Note that, while a single encryption or decryption stage lasts less than half a second, the PSNR of the reconstructed image is $44dB$, fairly higher than the $38dB$ threshold indicated for professional satisfaction [28]. the maximum deviation is $34dB$ in all cases.

Figure 3, 4, and 5 illustrate the results of using different values of N, A, and q. According to the complete visual similarity of the original images and the decrypted ones, we only visualize the original images in the following parts of this paper.

In Figure 3, in all cases $A=\frac{\pi}{2}$ and $q=4$ are used. The elapsed time and final quality are almost the same for all the images. Both the encryption and the decryption stages elapse almost equally $400ms$ in all cases and the $PSNR$ and maximum deviation values are $44dB$ and $34dB$, respectively. The key size varies from 5.5 bytes to 6148 bytes, resulting in a redundancy of 0.000699% to 0.782%. The chance to crack the code by a single trial changes from 0.00024 to less than 10^{-1470}. Hence, selecting a smaller N results in a much smaller code (and lower redundancy), but declines the security of the code. Also, N has no effect on the elapsed time and the quality of the reconstructed image. From this experiment and others we select $N=3$ as a proper choice.

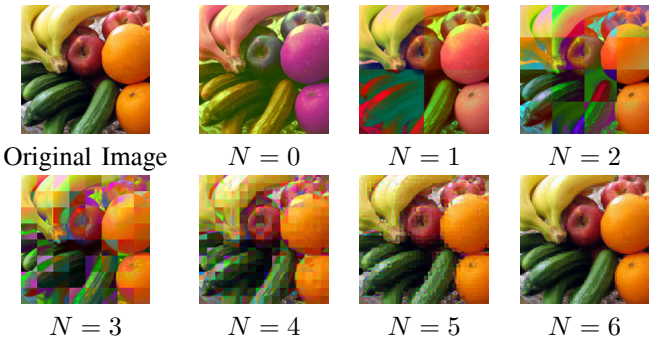

Fig. 3. Effect of the choice of N. Image courtesy of *Shohreh Tabatabaii Seifi* and *Ali Qanavati*.

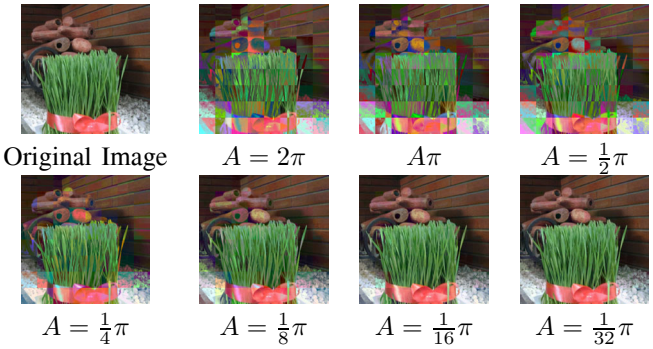

Fig. 4. Effect of the choice of A. Image courtesy of *Shohreh Tabatabaii Seifi* and *Ali Qanavati*.

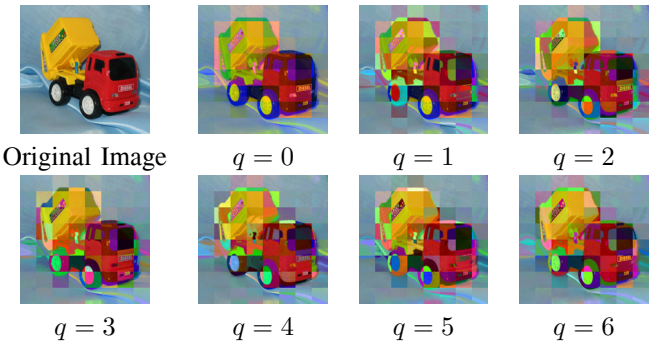

Fig. 5. Effect of the choice of q.

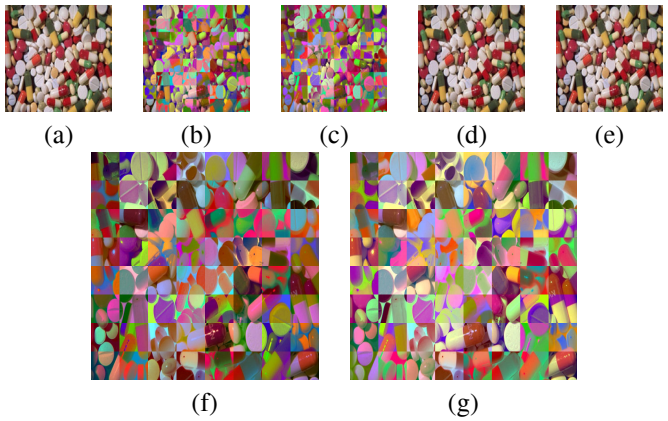

Fig. 6. Trial to decrypt an image using a key from a different encryption code of its own. (a) Original image. *Pills*, courtesy of *Karel de Gendre*. (b) and (c) Different encryptions. (d) and (e) Decryption using own keys. (f) and (g) Decryption using each other's keys.

In Figure 4, $N = 3$ and $q = 4$ with different values of A are used. As expected, elapsed time has no dependency on A ($t_e \simeq 420ms$ and $t_d = 340ms$). Though, the PSNR and maximum deviation depend on A. as A increases from 2π to $\frac{1}{32}\pi$, PSNR increases from $41dB$ to $47dB$ and maximum deviation increases from $26dB$ to $42dB$. This happens because smaller A results in less manipulations of data, resulting in less occurrences of numerical errors. As A lessens, the amount of visual distortions declines. In fact, when $A = \frac{1}{32}\pi$, there are literally no artifacts present in the encrypted image. As a compromise we use $A = \frac{1}{2}\pi$ in the rest.

In Figure 5, $N = 3$ and $A = \frac{1}{2}\pi$ with different values of q are used. As expected, the elapsed time does not depend on q ($t_e = 410ms$ and $t_d = 310ms$). Also, PSNR and maximum deviation are almost still (PSNR equals $43dB$ and maximum deviation equals $33dB$). But the key size varies from 4 bytes to 172 bytes and the probability of cracking the code by a single trial decreases from 1 to 3×10^{-410}. As a compromise we select $q = 4$.

Figure 6 illustrates a trial to decrypt an image using a key from a different encryption code of its own. In this way, Figure 6–a shows an image encrypted to the images shown in Figures 6–b and c. Now, decrypting the image with its own key results the images shown in Figures 6–d and e, while switching the keys results in the images shown in Figures 6–f and g. It is clear that while an image may be encrypted in $2^{3q2^{2N}}$ ways, no two encryptions share a similar key.

Figure 7 investigates the possibility of reproducing I_d from the compressed version of I_e. Figure 7–a shows a part of the original image and Figure 7–b and Figure 7–d illustrate the corresponding section of the results of the proposed decryption method when the encrypted image is, correctly, compressed using JPEG2000 and JPEG (without chrominance down–sampling). Here, JPEG2000 compression is utilized using *Advanced Batch Converter* 3.9 and JPEG compression is performed using *ACDSee* 5.0. In this way, JPEG2000 is utilized with rate of 90 and JPEG is performed with quality of 70. Figure 7–b shows the results of using the JPEG compression module incorporated into MATLAB, which performs the chrominance down–sampling. In this case, the quality of compression is selected 100. The fake lines are present when JPEG with chrominance down–sampling is used.

Figure 8 shows a sample utilization of the proposed method in the image sequence *Clair*. Each process of encryption or decryption of a single frame elapsed less than $100ms$ in this experiment and the PSNR was $44dB$ with maximum deviation of $34dB$.

IV. Conclusion

A new copy protection method for color visual objects is proposed. The method which is applicable for still images and

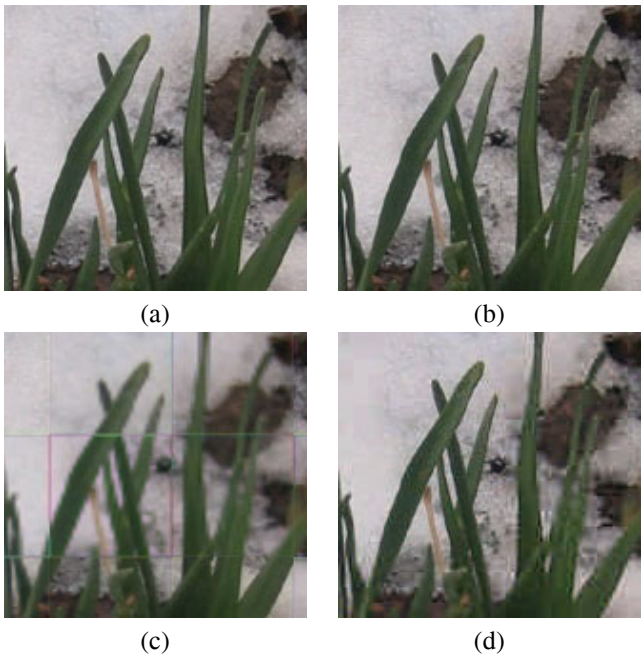

Fig. 7. Adaptivity of JPEG–based compression and the proposed method. a) Original image. b) JPEG2000. c) JPEG with chrominance down–sampling. d) JPEG without chrominance down–sampling.

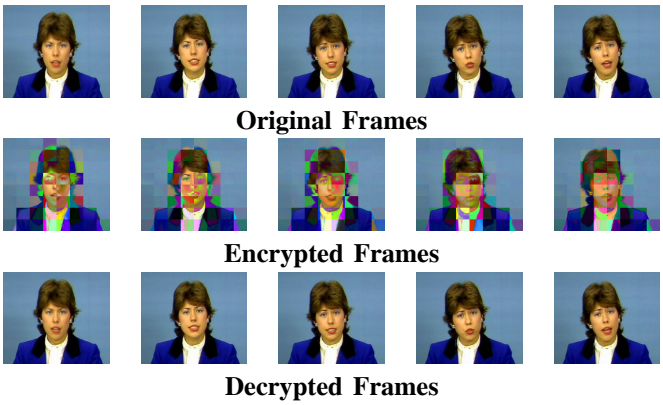

Fig. 8. Utilization of the proposed method in the image sequence *Clair*.

video in a realtime manner, is compatible with state–of–the–art compression techniques. The paper gives many examples of the utilization of the proposed method and shows that while the size of key over the image size is definitely small, the level of security given by the key is very high. Also, the degradation of the decrypted image is professionally desiring.

ACKNOWLEDGEMENT

This research is partially supported by *Iranian Telecommunication Research Center* (ITRC). The first author wishes to thank Ms. *Azadeh Yadollahi* for her encouragement and invaluable ideas.

REFERENCES

[1] M. Swanson, M. Kobayashi, and A. Tewfik, "Multimedia data–embedding and watermarking technologies," *IEEE*, vol. 86(6), pp. 1064–1087, 1998.

[2] J. Ruanaidh, W. Dowling, and F. Boland, "Watermarking digital images for copyright protection," *IEE Proceedings on Vision, Signal and Image Processing*, vol. 143(4), pp. 250–256, 1996.

[3] M. Kutter, F. Jordan, and F. Bossen, "Digital watermarking of color images using amplitude modulation," *Electronic Imaging*, vol. 7(2), pp. 326–332, 1998.

[4] P.-T. Yu, H.-H. Tasi, and J.-S. Lin, "Digital watermarking based on neural networks for color images," *Signal Processing*, vol. 81, pp. 663–671, 2001.

[5] C.-H. Chou and T.-L. Wu, "Embedding color watermarks in color images," *EURASIP Journal on Applied Signal Processing*, vol. 1, pp. 327–332, 2001.

[6] P. Tsai, Y.-C. Hu, and C.-C. Chang, "A color image watermarking scheme based on color quantization," *Signal Processing*, vol. 84, pp. 95–106, 2004.

[7] T.-Y. Chung, M.-S. Hong, Y.-N. Oh, D.-H. Shin, and S.-H. Park, "Digital watermarking for copyright protection of mpeg2 compressed video," *IEEE Transactions on Consumer Electronics*, vol. 44(3), pp. 895–901, 1998.

[8] C. Busch, W. Funk, and S. Wolthusen, "Digital watermarking: From concepts to real–time video applications," *IEEE Computer Graphics and Applications*, pp. 25–35, 1999.

[9] G. C. Langelaar, I. Setyawan, and R. L. Lagendijk, "Watermarking digital image and video data, a state–of–the–art overview," *IEEE Signal Processing Magazine*, pp. 20–46, 2000.

[10] G. Doerr and J.-L. Dugelay, "A guide tour of video watermarking," *Signal Processing: Image Coomunication*, vol. 18, pp. 263–282, 2003.

[11] D. Kundur and K. Marthik, "Video fingerprinting and encryption principles for digital rights management," *Proceedings of the IEEE*, vol. 92 (6), pp. 918–932, 2004.

[12] M. Kutter and F. A. P. Petitcolas, "A fair benchmark for image watermarking systems," in *Electronic Imaging'99, Security and Watermarking of Multimedia Content*, San Jose, Ca, USA, 1990.

[13] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Attacks on copyright marking systems," in *Information Hiding, Second International Workshop, IH98*, D. Aucsmith, Ed. Portland, Oregon, USA: Proceedings, LNCS 1525, Springer–Verlag, 1998, pp. 219–239.

[14] F. A. P. Petitcolas, "Watermarking schemes evaluation," *IEEE Signal Processing*, vol. 17 (5), pp. 58–64, 2000.

[15] ——, "Stirmark benchmark 4.0," http://www.petitcolas.net/ fabien/ watermarking/ evaluation/ index.html, Visited 2004.

[16] D. S. Wallach, "Copy protection technology is doomed," *IEEE Computer*, pp. 48–49, 2001.

[17] C. Herley, "Why watermarking is nonsense," *IEEE Signal Processing Magazine*, pp. 10–11, 2002.

[18] P. Moulin, "Comments on "why watermarking is nonsense"," *IEEE Signal Processing Magazine*, pp. 57–59, 2003.

[19] "What is the future for watermarking (part i)," *IEEE Signal Processing Magazine*, pp. 55–59, 2003.

[20] "What is the future for watermarking (part ii)," *IEEE Signal Processing Magazine*, pp. 53–57, 2003.

[21] S. A. Craver, M. Wu, and B. Liu, "What can we reasonably expect from watermarks?" in *2001 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, 2001, pp. 223–226.

[22] B. Grisworld, "Stopping webway rubbery: On–line image protection in the digital age," bgriswol@pop200.gsfc.nasa.goc, Visited 2004.

[23] C. Kim, "Content–based image copy detection," *Signal Processing: Image Communication*, vol. 18, pp. 169–184, 2003.

[24] A. Abadpour and S. Kasaei, "A new pca–based robust color image watermarking method," in *the 2nd IEEE Conference on Advancing Technology in the GCC: Challenges, and Solutions*, Manama, Bahrain, 2004.

[25] ——, "Color image processing (cip) dataset and toolbox for matlab," abadpour@math.sharif.edu, 2005.

[26] "Diopsis 740 dual–core dsp," www.atmel.com, 2004.

[27] "Tms320c6713, tms320c6713b floating–point digital signal processors," http://www.ti.com/, 2004.

[28] S. Katzenbeisser and A. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House Inc., 2000.