A Secure and Robust Digital Signature Scheme for JPEG2000 Image Authentication

Qibin Sun *Member, IEEE* and Shih-Fu Chang *Fellow, IEEE*

***Abstract*—In this paper we present a secure and robust content based digital signature scheme for verifying the authenticity of JPEG2000 images quantitatively, in terms of a unique concept named Lowest Authenticable Bit-Rates (LABR). Given a LABR, the authenticity of the watermarked JPEG2000 image will be protected as long as its final transcoded bit-rate is not less than the LABR. The whole scheme, which is extended from the crypto data-based digital signature scheme, mainly comprises signature generation / verification, Error Correction Coding (ECC) and watermark embedding / extracting. The invariant features, which are generated from fractionalized bit-planes during the procedure of EBCOT (Embedded Block Coding with Optimized Truncation) in JPEG2000, are coded and signed by the sender's private key to generate one crypto signature (hundreds of bits only) per image, regardless of the image size. Error Correction Coding (ECC) is employed to tame the perturbations of extracted features caused by processes such as transcoding. Watermarking only serves to store the check information of ECC. The proposed solution can be efficiently incorporated into the JPEG2000 codec (Part 1) and is also compatible with Public Key Infrastructure (PKI). After detailing the proposed solution, system performance on security as well as robustness will be evaluated.**

***Index Terms*—JPEG2000, Image Authentication, ECC, Digital Signature, Watermarking**

I. Introduction

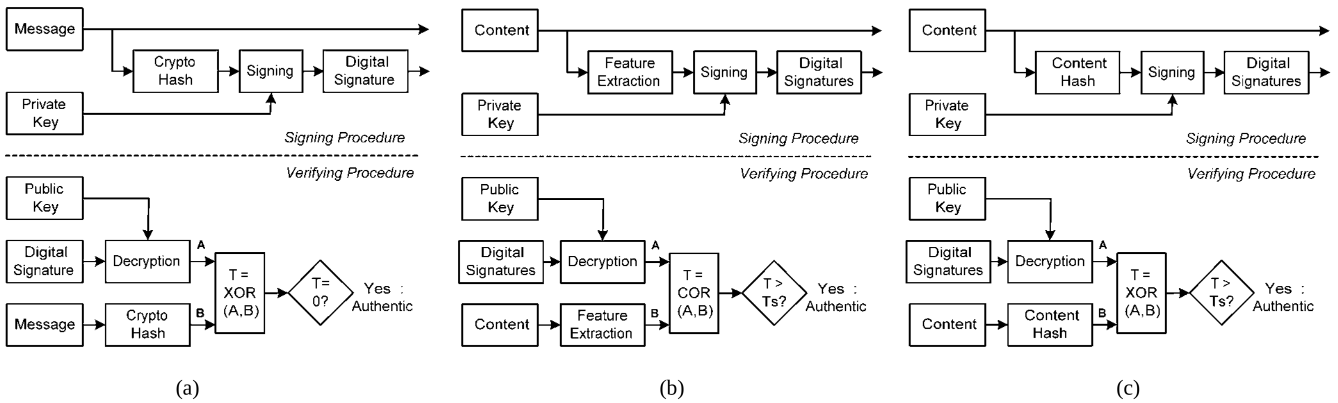
In these days more and more images are delivered over various public networks. We would like a digital signature scheme that allows two parties to exchange images while protecting the integrity of content and non-repudiation from the image sender. *Content integrity protection* means that the content isn't allowed to be modified in such a way that the content meaning is altered. *Sender's repudiation prevention* means that once an image sender generates the signature, he cannot subsequently deny such a signing if both the signature and the image have been verified as being authentic. The above scheme has been achieved in a fragile way (i.e., even one bit change in image is not allowable) either by crypto signature schemes such as RSA or DSA [1], or by public watermarking schemes [2]. However, the objective of this paper is to design such a scheme with the same functions but at semi-fragile (robust) level. The motivation stems from real applications where some image manipulations (e.g., lossy compression or transcoding) have to be considered allowable during the process of media transmission and storage (hereafter we refer to this type of distortion as *incidental distortion*) while other malicious modifications (e.g., image meaning alteration) from attackers should be rejected (hereafter we refer to this type of distortion as *intentional distortion*).

Consider the case of a police station transmitting a suspect's image to their officers' mobile terminals in order to identify a criminal. The policemen need to have guarantees that the image they received was indeed from the station. The police station also wishes to ensure that the sent image was not altered during transmission. However, due to different models of mobile terminals used by the policemen, and different bandwidth conditions between the station and each policeman, the authentic image will undergo some manipulations such as lossy compression or format conversion before reaching the policemen's mobile terminals. The image still needs to be securely authenticated in such a case. Therefore, we have two requirements to consider. On the one hand, the scheme must be *secure* enough to prevent any attacked image from passing the authentication (sensitive to intentional distortion). On the other hand, the scheme also needs to be *robust* enough to accept an image that has undergone some acceptable manipulations (insensitive to incidental distortion).

It is worth noting that prior works on semi-fragile watermarking solutions work well on protecting content integrity [3]. However, they cannot solve the problem of preventing signing repudiation from the sender because most watermarking approaches share the same key both for watermark embedding and for watermark extraction. Therefore, in this paper our survey on prior related works only focus on semi-fragile digital signature based solutions.

A. Prior work on semi-fragile signature based authentication

Fig. 1(a) shows the brief diagram of crypto /digital signature [1]. Given a message of arbitrary length, a short fixed-length digest is obtained by a crypto hash operation (e.g., 128 bits by MD5 or 160 bits by SHA-1). The signature is generated by using the sender's private key to sign on the hashed digest. The original message associated with its signature is then sent to the intended recipients. The recipient can verify a) whether his received message was altered, and b) whether the message was really sent from the sender, by using the sender's public key to authenticate the validity of the attached signature. Based on security consideration, the requirements for a crypto hash functions are [1]: a). Given a message m and a hash function H, it should be easy and fast to compute the hash $h = H(m)$. b). Given h, it is hard to compute m such that $h = H(m)$ (i.e., the hash function should be one-way). c). Given m, it is hard to find another data m' such that $H(m') = H(m)$ (i.e., collision free). The final authentication result is then drawn from a bit-



(a) (b) (c)

Fig. 1. Digital signature schemes for message / content authentication. (a). The digital signature scheme for message authentication. (b). Content authentication based on correlation of extracted features. (c). Content authentication based on hamming distance between content hashes.

to-bit comparison between two hash codes (Refer to Fig. 1(a), one is decrypted from the signature and the other is obtained by re-hashing the received message) by the criterion: Even if one bit difference exists then the received message will be deemed unauthentic.

Originating from the ideas of *fragile* digital signature as described above, Chang [4] and other researchers proposed to use some typical content-based measures as the selected features for generating content signature, by assuming that those features are insensitive to incidental distortions but sensitive to intentional distortions. The features include histogram map, edge/corner map, moments, and more. Considering that applying lossy compression such as JPEG should be deemed an acceptable manipulation in most applications, Lin and Chang [5] discovered a mathematical invariant relationship between two coefficients in a block pair before and after JPEG compression and selected it as the feature. Similarly, Lu and Liao [6] presented a structural signature solution for image authentication by identifying the stable relationship between a parent-child pair of coefficients in the wavelet domain. Referring to Fig. 1(b), we can see that the module of “crypto hash” in Fig. 1(a) has been replaced with the module of “feature extraction” in order to tolerate some incidental distortions. The replacement is applied because the acceptable manipulations will cause changes on the content features, though the changes may be small compared to content-altering attacks. Such “allowable” changes on the content features make the features non-crypto-hashing. (Any minor changes to the features may cause a significant difference in the hashed code due to the nature of the crypto hashing). Accordingly, as a result of the incompatibility with crypto hashing, the generated signature size is proportional to the size of the content, which is usually very large. Because in typical digital signature algorithms the signature signing is more computational than the signature verifying, this will also result in a time-consuming signing process (The formed signature whose size is much greater than 320 bits [7] needs to be broken into small pieces (less than 320 bits) for signing). Furthermore, no crypto hashing on content features will make the decision of authenticity usually based on comparison of the feature distance (between the original one decrypted from the signature and the one extracted from the received image) against a threshold value which is hard to determine in practice and bring some potential security risks.

Since directly applying crypto hashing to images (features) seems infeasible and feature-based correlation approaches still do not resolve issues such as signature size and security risks, other researchers have already been working on designing their own hash functions named *robust hash* or *content hash*, see Fig. 1(c). Differing from crypto hash functions, content hashing aims to generate two hash codes with short Hamming distance if one image is a corrupted version of another image by incidental distortions. In other words, if two images or two image blocks are visually similar, their corresponding hash codes should be close in terms of Hamming distance. For example, in Fridrich and Goljan's [8] solution, the hash function is designed to return 50 bits for each 64x64 image block by projecting this 64x64 image block onto a set of 50 orthogonal random patterns with the same size (i.e., 64x64) generated by a secret key. The final hash value of the image is obtained by concatenating the hash codes from all blocks.

Xie, Arce and Graveman [9] proposed a content hash solution for image authentication: Approximate Message Authentication Codes (AMAC). The AMAC is actually a probabilistic checksum calculated by applying a series of random XOR operations followed by two rounds of majority voting to a given message. The similarity between two messages can be measured by the Hamming distance of their AMACs. The length of an AMAC is typically around 80-400 bits. Venkatesan, Koon, Jakubowski and Moulin [10] also proposed a solution for generating content hash for image. Firstly, the image is randomly tiled and wavelet transform is applied to each tile independently. Some statistics measures such as mean and variance are calculated in each wavelet domain. Secondly those obtained measures are quantized using a random quantizer (i.e., the quantization step size is random) to increase security against attacks. Thirdly, the quantized statistics measures in each tile are decoded by a pre-defined ECC scheme. Finally the content hash value of the image is obtained by concatenating the ECC decoder outputs of all tiles.

However, some limitations also exist for this type of content hash based schemes. Due to a short representation of generated content hash, it is very hard for the content hash itself to differentiate incidental distortions from intentional distortions, especially when the intentional distortions are the results of attacks to only part of the image. Consequently, it is very difficult to set a proper threshold for making the authentication decision. Furthermore, this scheme also lacks the ability to locate the content modifications if the authentication fails.

We can summarize that, in order to both protect content integrity and prevent sender's repudiation, a good semi-fragile digital signature scheme should satisfy the following requirements. Firstly, the size of the signature signed by a semi-fragile scheme should be small enough to reduce the computational complexity. Ideally, the size is comparable to or the same as that which is signed by a fragile crypto signature scheme. Secondly, such a scheme should be able to locate the attacked portion of the image because that would easily prove the authentication result. Lastly and most importantly, a proper content-based invariant feature measure should be selected in such a way that it is sensitive to malicious attacks (intentional distortions) and robust to acceptable manipulations (incidental distortions). In addition, the feature should characterize the local property of an image because most attacks may only act on part of the image (e.g., only changing the some digits in a cheque image); Feature selection is actually application dependent: Different applications have different definitions of incidental distortions as well as intentional distortions. Therefore, defining acceptable manipulations and unallowable modifications of the image content is the first step in designing a good semi-fragile authentication system.

B. JPEG2000 and its demands for authentication

Recently the Joint Photographic Experts Group issued a new international image compression standard: JPEG2000 [11,12]. Compared to DCT-based JPEG standard, this new standard employs Wavelet Transform (WT) to obtain better energy compaction for the image. Therefore, it can achieve improved compression efficiency, especially on the CBR lower than 0.25bpp. By adopting multi-resolution representation and embedded bit-stream, decoding can be flexibly achieved in several progressive ways such as SNR scalability and resolution scalability. Some other rich features in JPEG-2000 include Region-of-interest (ROI) coding, error resilience, random codestream access, etc [11, 12, 13]. Fig. 2 illustrates the main encoding procedure of JPEG2000 (Part 1). Given a color raw image, after forward multi-component transformation (i.e. color transformation, *Module 1*), it will be decomposed into different resolution levels and subbands by forward wavelet transformation (*Module 2*). It is then quantized by a uniform scalar quantizer (*Module 3*). For some applications, Region-of-Interest (ROI) coding may be applied (*Module 4*). An adaptive binary arithmetic coder will start encoding all quantized WT coefficients from the MSB bit-plane to the LSB bit-plane. The final bit stream will depend on the pre-defined compression bit-rate (CBR) and progression order. The last two steps are also called EBCOT (*Module 5*). By EBCOT, the user can quantitatively define the intended compression ratio for his images in terms of bit rate. The JPEG2000 file stream is finally formed (*Module 6*).

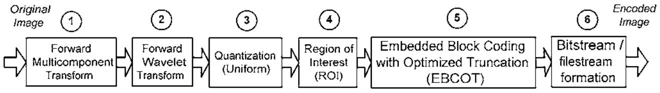

Original Image
1
Forward Multicomponent Transform
2
Forward Wavelet Transform
3
Quantization (Uniform)
4
Region of Interest (ROI)
5
Embedded Block Coding with Optimized Truncation (EBCOT)
6
Bitstream / filestream formation
Encoded Image

Fig. 2. A brief JPEG2000 (Part 1) encoding diagram

In the procedure of JPEG2000 coding, EBCOT plays a key role in the bit-rate control. Refer to Fig. 3, the image is decomposed into 3 resolution levels (1LL/1LH/1HL/1HH, 2LH/2HL/2HH and 3LH/3HL/3HH). WT coefficients in higher resolution levels (e.g., 1LL) are quantized in a finer way, i.e., the WT coefficients in the higher resolution level are represented by more bit-planes. The quantized coefficients in each subband are partitioned into small rectangular blocks (64x64 except image boundary). These blocks are referred to as codeblocks. Each codeblock is encoded independently. The encoding procedure in each codeblock is comprised of two steps [14]. Firstly, each bit-plane is fractionalized into 3 sub-bit-planes (i.e., Significant Pass, Magnitude Refinement Pass and Clean-up Pass) based on a pre-defined look-up-table modeling the contextual information of its neighboring bits. We refer to these sub-bit-planes as *fractionalized bit-planes.* Secondly, each fractionalized bit-plane is coded to form the bit-stream by an adaptive binary arithmetic coder. Let $\{B_i\}_{i=1,2,...}$ denote the set of all codeblocks which represent the image [14]. For each codeblock, B_i, a separate bit-stream is

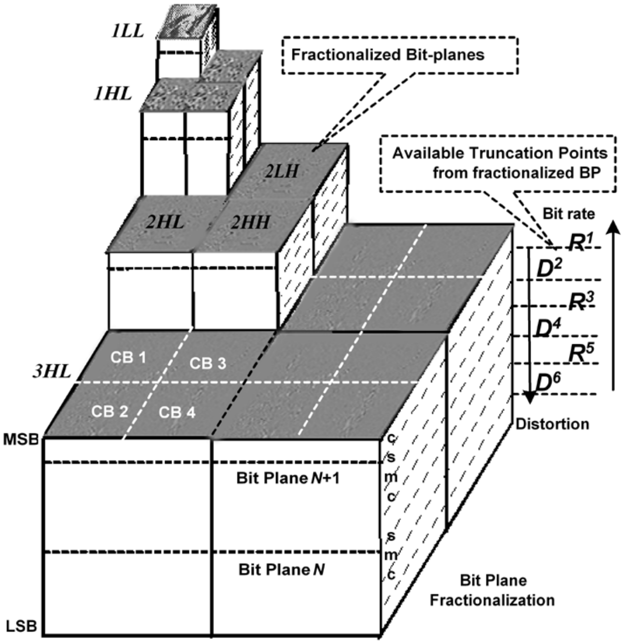

1LL
1HL
Fractionalized Bit-planes
2LH
Available Truncation Points from fractionalized BP
2HL
2HH
Bit rate
R1
D2
R3
D4
R5
D6
Distortion
3HL
CB 1
CB 3
CB 2
CB 4
MSB
c
s
m
c
s
m
c
Bit Plane N+1
Bit Plane N
LSB
Bit Plane Fractionalization

Fig. 3. Illustration on the principle of EBCOT.
Original image (256x256) is decomposed into 3 levels by wavelet transform: 1LL/1LH/1HL/1HH, 2LH/2HL/2HH, 3LH/3HL/3HH. The level 3 (rougher) has fewer bit-planes than the other levels. Each bit-plane is further fractionalized into 3 sub-bit-planes (referred as **S**ignificant pass, **M**agnitude refinement pass and **C**lean-up pass) except the MSB (only one: clean-up pass). R-D summary is drawn in terms of available truncation points derived from bit-plane fractionalization.

generated. Due to fractionalization of the bit-plane, the generated bit-stream can be truncated to a variety of discrete lengths, $R_i^1, R_i^2, R_i^3, \ldots$, and the distortions incurred when reconstructing from each of the truncated subsets is estimated and denoted by $D_i^1, D_i^2, D_i^3, \ldots$, here R_i^0 and D_i^0 are the rate and distortion assuming all fractionalized bit-planes are dropped. With more fractionalized bit-planes are included, R_i'' will increase while D_i'' will decrease. During encoding process, R_i'' and D_i'' are computed and stored in a compact form with the compressed bit-stream itself. Once the entire image has been compressed codeblock by codeblock, a post-processing operation scans over all the compressed codeblocks and determines the extent to which each codeblock's embedded bit-stream should be truncated in order to achieve a particular global targeted CBR or distortion bounds. The final bit-stream is then formed by concatenating the coded codeblocks together in a pre-defined progression order, together with information to identify the number of bytes, R_i'' , which are used to represent each codeblocks [11,14].

Because of its novel features, JPEG2000 has received significant attention and support from various applications such as mobile communication, medical imaging, digital photography, and digital library. Such applications demand content integrity protection and sender repudiation prevention, i.e., a secure and robust authentication solution for JPEG2000 images. However, flexible and efficient coding strategies in JPEG2000 also pose some new challenges. For example, JPEG2000 is known as "encode once and decode many times". This means that once a JPEG2000 codestream is formed at a given CBR, any other new codestream whose bit-rate is lower than the given CBR can be simply obtained by just truncating this formed codestream from its end. Hence it requires that the proposed authentication solution should be able to align with such a coding flexibility.

In this paper, we present an integrated content-based authentication scheme targeting at verifying the authenticity of JPEG2000 images quantitatively and securely in terms of a unique concept named lowest authentication bit-rates (LABR). Given a LABR which is similar to the targeted CBR used in the JPEG2000 image, the authenticity of the watermarked JPEG2000 image will be protected as long as its final transcoded bit-rate is not less than the LABR. The whole scheme, which is an extension of crypto data-based signature schemes, is mainly comprised of three modules: signature generation / verification, ECC and watermark embedding / extraction. The invariant signatures, which are generated from fractionalized bit-planes during the procedure of EBCOT, are coded and signed by the sender's private key to generate one crypto signature per image only. ECC is employed to tame the perturbations of extracted features caused by procedures such as transcoding. Instead of sending original data associated with the signature to the recipients, we send them the watermarked image associated with its content-based crypto signature. Differing from other watermarking-only approaches for content integrity protection where secure embedding is required, watermarking in our proposed scheme only functions to explicitly store the check information[*] of ECC. The proposed solution can be fully and efficiently incorporated into the procedures of JPEG2000 encoding/decoding and PKI.

The rest of this paper is organized as follows. Section II presents a brief overview of the proposed solution starting from defining acceptable manipulations. In section III, we describe our proposed schemes in details, mainly focusing on feature extraction, signature generation and verification, ECC and watermarking strategies etc, all of which are based on LABR. Security analysis, experimental methodology and results are then discussed in Section IV for the purpose of system performance evaluation. Conclusions and future work will be given in Section V.

II. Overview of Proposed Solution

As we described in the previous section, a properly selected feature is very important for system security as well as robustness. The feature should be sensitive to any intentional distortions while insensitive to all incidental distortions. However, feature selection is application dependent. A good feature for one application isn't necessarily good for another. In other words, only after the application is specified can we analyze which content manipulations are acceptable and which content modifications are not allowable. Only after the acceptable manipulations and unallowable modifications are defined can we evaluate which feature is the best.

A. Targeted acceptable manipulations

Currently JPEG2000 committee [11] is calling for proposal for solving the security issues (Part 8: JPSEC). In the proposal, they clearly indicated that they want the security solutions (e.g., authentication) at the content level. Based on some JPEG2000 target applications such as adaptive image transmission, we identify a list of acceptable content-based manipulations for our proposed solution. Assuming a JPEG2000 image is signed by our scheme, its authenticity could be still protected if it is recompressed, transcoded, or watermarked.

- Recompression

The terminals request a JPEG2000 image from a server via some routes. In order to reduce the computation burden of the server, the routes may share some transcoding tasks to meet the needs of terminals with different capacities. Typically the transcoding involves decoding-and-encoding the image. The re-encoding may be repeated multiple times. Such manipulations will unavoidably introduce some incidental distortions. Fig. 4(a) shows the incidental distortions introduced during multi-cycle compression. The original image repeatedly undergoes JPEG2000 compression, i.e., compressing the original image into 1bpp and decompressing it to an 8-bit raw

[*] In this paper, we consider a special type of ECC codes called systematic codes. Given a systematic (*n*, *k*) code, the first *k* components in *n* are the same

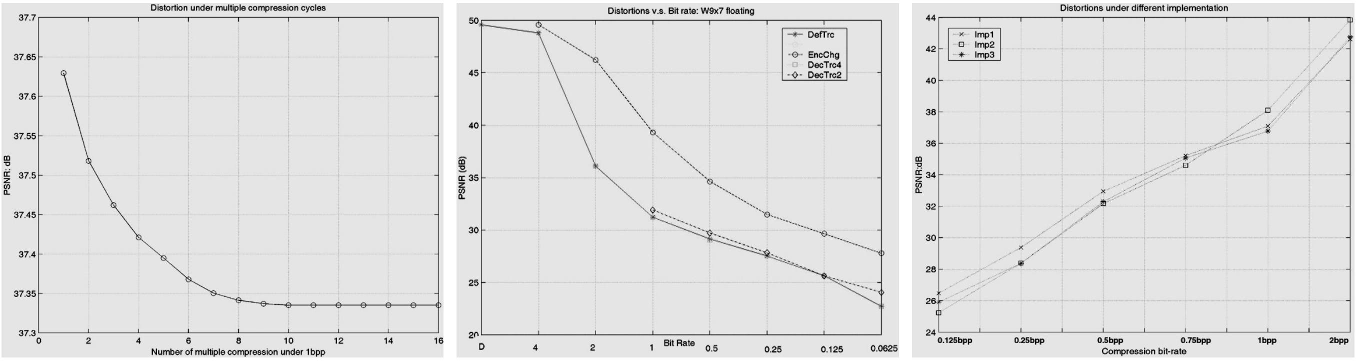


Fig. 4. Some examples of incidental distortions

(a) Distortions as a function of multiple compression cycles. The testing procedure is: Compress original image into 1bpp, decompress it to raw image, compress it again at 1bpp……measure each PSNR with reference to the original image. (b) Distortions as a function of different JPEG2000 coding / transcoding means. DefTrc, DecTrc4 and DecTrc2 refer to compressing image with full bit-rate, 4bpp, 2bpp respectively and then trancoding to the testing bit-rates. EncChg refers to directly compressing the original image to the testing bit-rates. (c) Distortions as a function of different implementations with reference to originals. Among three kinds of JPEG2000 implementations, Imp1 is based on Java and the other two are based on C++.

image (one cycle), compressing it again at 1bpp, ... The result shows the PSNR with reference to the original image. Furthermore, JPEG2000 provides several flexible ways in obtaining a compressed image with its targeted bit rate either by directly compressing from a raw image, or by truncating / parsing from a compressed codestream. Such different coding / transcoding approaches also cause the incidental distortion, as shown in Fig. 4(b). DefTrc, DecTrc4 and DecTrc2 refer to compressing the original image with full, 4bpp and 2bpp respectively and then truncating to the targeted bit-rate. EncChg refers to directly compressing the original image to the targeted bit-rate. The result PSNR of the compressed image with reference to the original image tells us that the introduced incidental distortion is closely related to selected coding /transcoding means.

- Format / Codec Variations

Differences may exist between different implementations of JPEG2000 codec by different product providers. Such differences can be due to different accuracies of representation in the domains of (quantized) WT, color transformation, and pixel. Fig. 4(c) demonstrates such incidental distortions caused by different implementations in terms of PSNR with reference to the original image. Imp1 is based on the JJ2000 implementation [12], a java version JPEG2000 implementation, while imp2 and imp3 are other two common C++ versions implementations: one is from HP [13] and the other is Jasper [11].

- Watermarking

Image data is “manipulated” when authentication feature codes are embedded back into the image. Such a manipulation should not cause the resulting image to be unauthentic.

B. System brief description

Our proposed signing procedure for semi-fragile content-based image authentication is shown in Fig. 5. In the scheme, signature generation/verification modules are employed for image signing and authentication. Watermark embedding / extraction modules are only used for storing ECC check information. Instead of directly sending an original image to recipients, we only pass them the watermarked copy associated with one signed digital signature whose length is usually very short (e.g., if we adopt a signing algorithm in [7], the signature size is only around 320 bits regardless of the original image size). Refer to Fig. 5, among four inputs: original image, JPEG2000 target CBR (CBR) *b*, LABR *a* and the image sender's private key, CBR is the mandatory input for compressing images into JPEG2000 format. In addition to the original image, only two inputs are needed to generate the JPEG2000 image signature in a content-based way: the private key and the LABR *a*. If a JPEG2000 image signature is generated with LABR value *a*, a new image with CBR *b* will be authentic as long as *b* is greater than *a* and the new image is derived from defined acceptable manipulations or transcoded (by parsing or truncation) from a compressed JPEG2000 image. The original image undergoes color transformation, wavelet transformation, quantization, and ROI (*module 1-4*), which are all basic procedures in JPEG2000 encoding [11]. EBCOT (*module 5*) is employed for bit plane fractionalizing /encoding and optimal bit rate control. We extract content-based features (*module 6*) from the available fractionalized bit planes by assuming the image is encoded above LABR (i.e., *b*>*a*). Details of the content-based feature extraction from EBCOT will be described later. Feature values from each codeblock are thresholded and ECC coded (*module 7*) to generate corresponding parity check bits (PCBs, *module 8*). Then we take PCBs as the seed to form the block-based watermark (*module 9*). One necessary condition in watermarking is that the embedded watermark should be robust enough for extraction from received images under acceptable manipulations. Since incidental changes to the embedded watermarks might occur, we apply another ECC scheme again to encode the PCB data before they are embedded. The watermark data for each block are embedded into either the same block or a different block. The watermark embedding location is also determined based on the LABR value. Note only the PCB data (not including the feature codes) are embedded in the above watermarking process. All codewords

as the original message and called information symbols. The remaining *n-k* components are called redundancy or check symbols (check information).

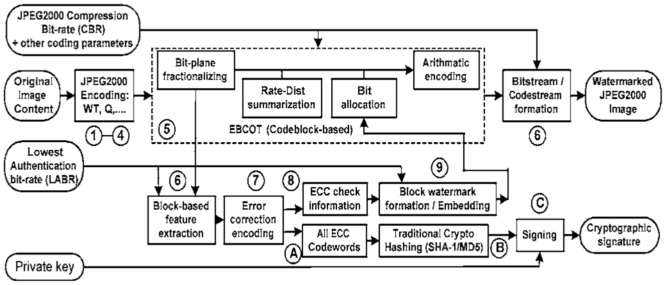

JPEG2000 Compression
Bit-rate (CBR)
+ other coding parameters
Original Image Content
JPEG2000 Encoding: WT, Q,....
1
4
Bit-plane fractionalizing
Rate-Dist summarization
Bit allocation
Arithmatic encoding
5
EBCOT (Codeblock-based)
Bitstream / Codestream formation
6
Watermarked JPEG2000 Image
Lowest Authentication bit-rate (LABR)
6
7
8
9
Block-based feature extraction
Error correction encoding
ECC check information
Block watermark formation / Embedding
All ECC Codewords
A
Traditional Crypto Hashing (SHA-1/MD5)
B
C
Signing
Cryptographic signature
Private key

Fig. 5. Integrated framework for content-based authentication (Signing)

(features together with their corresponding PCBs, *module A*) from all resolution levels and all subbands are concatenated and the resulting bit sequence is hashed by a crypto hashing function such as MD5 or SHA-1 (*module B*). The generated semi-fragile hash value can then be signed using the image sender's private key to form the crypto signature (*module C*), based on traditional crypto signature schemes such as RSA or DSA. Differing from these traditional data-based signature scheme in which the original data are sent to the recipients associated with its signature, our proposed solution sends out the watermarked image to the recipients.

Refer to Fig. 6, to authenticate received image, in addition to the image itself, two other pieces of information are needed: the signature associated with the image (transmitted through external channels or as embedded watermarks), and the image sender's public key. The image is processed in the same way as content signing (*module 1-4*): decompose and quantize image into blocks, to extract features for each block. (Note that here we assume the JPEG2000 image has been decoded into raw image and we authenticate this raw image given LABR. If the image is still JPEG2000 compressed, the features and watermarks can also be obtained from the JPEG2000 decoding procedure). From those embedded watermarks, we extract the PCB data generated at the source site (*module E*). Note that the features are computed from the received image (*module 6*), while the PCB data are recovered from the watermarks that are generated and embedded at the source site (*module F*). After we combine the features and the corresponding PCBs to form codewords (*module D*), the whole image verification decision could be made orderly. First, we calculate the syndrome of the codeword block by block to see whether any blocks exist whose codewords are uncorrectable. If any exists, then we claim the image is unauthentic and use the above ECC checking process to display the possible alteration locations. If all codewords are correctable (i.e. errors in any feature code are correctable by its PCB), we repeat the same process as the source site: concatenate all corrected codewords into a global sequence (*module A*) and cryptographically hash the result sequence (*module B*). The final verification result is then concluded through a bit-by-bit comparison between these two hash sets (i.e., one is this newly generated (*module B*) and the other is decrypted from the associated signature (*module G*) by the obtained public key): if any single bit differs, the verifier will deem the image unacceptable ("unauthentic"). The detailed descriptions on feature selection, signature generation, ECC and watermarking will be given in the next section.

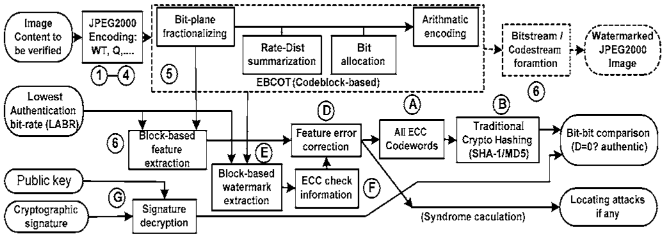


Image Content to be verified
JPEG2000 Encoding: WT, Q,....
1
4
Bit-plane fractionalizing
Rate-Dist summarization
Bit allocation
Arithmatic encoding
5
EBCOT(Codeblock-based)
Bitstream / Codestream foramtion
Watermarked JPEG2000 Image
6
Lowest Authentication bit-rate (LABR)
6
Block-based feature extraction
D
Feature error correction
A
All ECC Codewords
B
Traditional Crypto Hashing (SHA-1/MD5)
Bit-bit comparison (D=0? authentic)
E
Block-based watermark extraction
ECC check information
F
Public key
G
Cryptographic signature
Signature decryption
(Syndrome caculation)
Locating attacks if any

Fig. 6. Integrated framework for content-based authentication (Verifying)

III. LABR Based Signature Generation and Distortion Bounded Watermarking

A. Bit rate allocation and distortion estimation in EBCOT

As described in the previous section, EBCOT provides a finer scalability of image content resulting from multiple-pass encoding on the codeblock bit-planes (i.e., fractionalizing each bit-plane into 3 sub-bit-planes: Significant Pass, Magnitude-refinement Pass and Clean-up Pass based on pre-defined contextual models [14]). The coding summaries for each code-block after EBCOT include feasible truncation points, their corresponding compressed size (rate *R*), and estimated distortions (*D*). The target CBR is achieved by globally scanning the contributions from all codeblocks and optimally selecting truncating points for each codeblock on the formed R-D curve (using Lagrange Multiplier method).

In our solution, we choose to extract robust features from the EBCOT based on the following considerations. Firstly, EBCOT is the last processing unit prior to forming the final compressed bitstream in JPEG2000. It means all possible distortions have been introduced before running EBCOT while no distortion is introduced afterwards. If we are to authenticate the encoding output or the output from directly truncating or parsing the compressed bitstream, the extracted features will not be distorted. Secondly, EBCOT expands the scalability of an image by fractionalizing the bit-plane. The fractionalized bit planes of EBCOT represent closely the image content, and thus to alter the image while intentionally keep the same fractionalized bit planes is difficult. Thirdly, in JPEG2000, EBCOT is the engine of bit rate control and provides exact information about specific fractionalized bit-planes of data to be included in the final bit stream given a target CBR. Such information allows us to specify the invariant layers of data and quantitatively select an authenticable level.

One important property is worth noting here: Three passes included in the bit stream at a rate (say *a*) are always included in the bit stream at a higher rate (>=*a*). This property is illustrated in Fig. 7, in which no lines cross other lines corresponding to the passes included at different CBRs. Such a property is important in our scheme for obtaining invariant feature sets. Fig. 7 shows all available truncation points (i.e., cutting layers) in all code-blocks of an image with size 128x128 during its encoding procedure (actually each subband only has one codeblock with the maximum size of 64x64). The

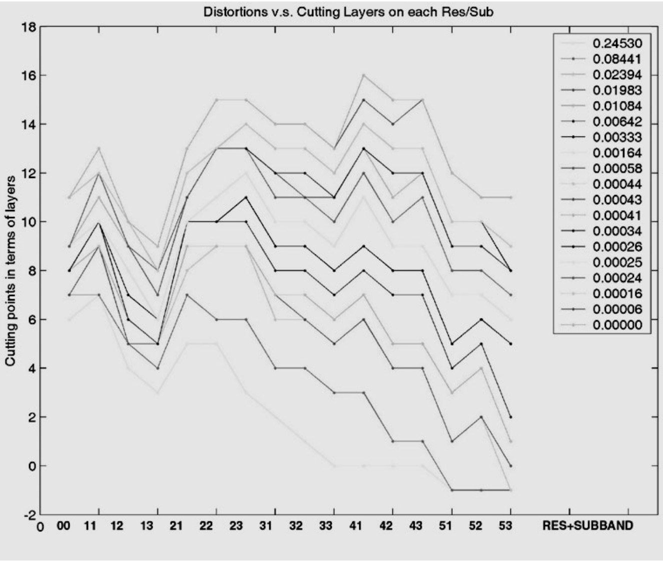


Fig. 7. Testing result on multiple cutting layers vs. bit-rate on 128x128 Lena image: 00: 0LL, 11: 1LH, 12: 1HL, 13: 1HH…..53: 5HH (one codeblock per subband). Lines from top to bottom correspond to the bit-rates from bottom to top shown on the right side, we can see that no cross cutting lines among different layers (overlapping could happen). Note that layer –1 means not selecting from it for codestream formation.

image is decomposed into 5 levels: 00 means the lowest LL band, 11 means 1LH band, ... , and 53 means 5HH band. We can see that the no cutting lines (linking all truncation points along all levels/subbands) cross each other (although there are some overlapping lines). Note that lines from top to bottom correspond to the available CBRs from bottom to top shown on the right of Fig. 7. Although a little perturbation could still occur during the acceptable manipulations, this will be tackled by adopting ECC techniques.

EBCOT also provides a convenient and fast computation mechanism for estimating the associated distortions of truncating selected passes of each codeblock. Refer to [14], this computation can be performed with the aid of two small lookup tables which do not depend on the coding pass, bit-plane or subband involved. For example, in a codeblock with all *Z* bit-planes included (*Z*-1 refers to the MSB and 0 refers to the LSB), its distortion can obviously be thought of as a constant value which may readily be computed from the Wavelet filter kernel and the quantization step size in the corresponding subband [14]. We omit the details here for the reason of simplicity. Later, suppose we need to skip the p^{th} bit-plane to achieve our targeted CBR, which means we are setting all sample bits below the p^{th} bit-plane to zero. Its total resulted distortion can be estimated as follows.

$$Distortion = \sum_{i=0}^{p} 2^{2i} \Delta^2 \cdot f\left(v_i'[m,n]\right) \quad (1)$$

where $v_i'[m,n] = 2^{-i} v[m,n] - 2\left\lfloor \frac{2^{-i} v[m,n]}{2} \right\rfloor$, and $v[m,n]$ is the magnitude of the quantized WT coefficient at location [*m*, *n*]. Δ^2 denotes the contribution of distortion in the reconstructed image which would result from an error of exactly one step size in a single coefficient. Note that *f* also depends on the current state (significant or not) of this coefficient. We can see that such a mechanism could allow us to quantitatively control the image quality degradation caused by truncation as well as watermark embedding, because it conveniently provides us explicit information on how much image quality degradation we should pay for to gain targeted compression ratio or robustness of embedded watermarks.

B. LABR based feature extraction and its invariance to incidental distortions

We know that given a targeted CBR, the truncation point in each codeblock can be located in terms of its fractionalized bit-planes, by running EBCOT (*module 5* in Fig. 5 and 6). Based on the above observation, the selected features for signature generation must come from those bit-planes which will be included into the final bit-stream, assuming that the targeted CBR is LABR (remember that the actual targeted CBR should be greater than LABR). We select two measures as invariant features which can be directly obtained from the procedure of EBCOT: one is the state of passes (i.e., the fractionalized bit planes) of MSBs, and the other is the estimated distortion associated with each pass computed based on (1). Intuitively, taking the states of passes as the features is quite straightforward: it is actually a layer of MSB bit-planes in a codeblock (part of image content) and already represented in binary form. The reason why we select the estimated distortions as another feature rather than directly stringing all passes is based on the following observation: Within one codeblock saying 64x64, the number of possible corrupted bits located at LSBs' bit-planes will be greater than those located at MSBs'. From (1), the estimated distortion is actually a measure of the change of "1" in a given bit-plane. It means that when we estimate the distortions, the changes from MSBs should be assigned higher weights than those from LSBs. In other words, the importance of changing MSBs is greater than that of changing LSBs because corrupting MSBs is more visible than corrupting LSBs. (This is what EBCOT does). Therefore, adopting estimated distortions among different bit-planes is more reasonable than directly stringing all bit-planes (i.e., different passes). By combining the estimated distortions with the states of passes at MSBs, the entire content specified by LABR is protected in 3-dimensions (2-D from states of passes and 1D from estimated distortions, as shown in Fig. 3).

We have conducted various experiments to test the invariant properties under acceptable manipulations as defined before. Fig. 8 shows an example of differences among Significant Passes (SP) of one codeblock from different codec implementations. The test software is obtained from [11, 12, 13] where JO means SP of the original image based on an encoder implemented in Java, CO represents the difference between SP from two different encoders (Java vs. C++), JOC and COC represent the differences between JO and SP after multiple compression cycles (Java vs. C++) respectively, JA and CA represent the differences between JO and SP of attacked image (Java vs. C++), JAC and CAC represent the differences between JO and SP of attacked image plus multiple

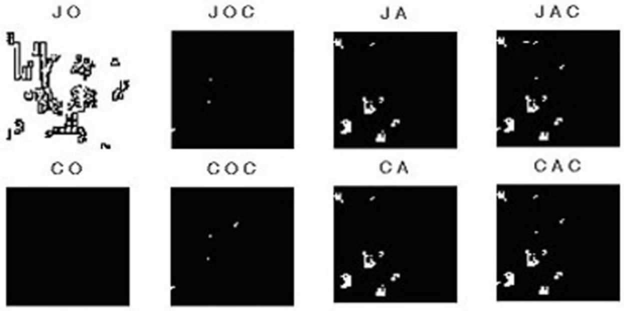


Fig. 8. Differences between the states of significant pass (SP) of the test images (one codeblock only with the size of 64x64). The shown SP is the fractionalized results of the second highest bit-plane.

compression cycles (Java vs. C++). As we expected, codec implementation variations and acceptable manipulations introduced much less changes to the bit plane than content-altering attack. Similarly, as shown in Fig. 9, estimated distortions associated with the different passes of EBCOT are much more stable during acceptable manipulations than attacks. However, as seen from Fig. 8 and 9, some small perturbations are still introduced into the features. This is the reason we employ ECC to tame such distortions in order to generate a stable crypto signature. Note that the extracted content-based features comprise a set of binary vectors which combines the states of passes at MSBs (in binary representation) and the thresholded estimated distortions.

C. ECC for taming incidental distortions

Our interest in ECC stems from the fact that extracted features used for signature generation and verification cannot be modeled and measured perfectly [15]. Each extracted feature set results in a vector that is always at some Hamming distance from another feature set under the acceptable manipulations. Hence error correction is critical in forming stable crypto hash value and digital signature under incidental distortions. On the other hand, the capability of correcting errors has to be carefully defined in order to differentiate incidental distortions from intentional distortions in the case of wrongly judging some attacked images to be authentic.

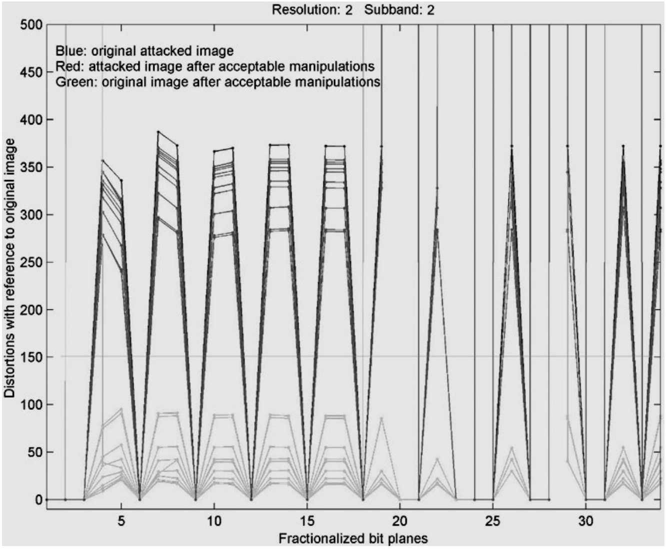


Fig. 9. The distances of distortions between original and attached images under various manipulations. (A codeblock in Res 2, Sub: HL).

The estimated distortions for various versions of images are parsed. The tested acceptable manipulations shown in the figure include multi-cycle compression, various parsing /truncation transcoding, different codec, JPEG etc. Note that the tested bit-rate shown in this figure is on 0.8bpp, therefore the fractionalized bit-planes after 17 are not included into final code-stream and will not affect the content verification.

A (N, K, D) code [16] is a code of N bit codewords where K is the number of message digits (the extracted feature set in the paper) and D is the minimum distance between codewords. N - K = P is the number of parity check digits (i.e., checksum or check information). Note that an error correcting code with rate K/N can correct up to T = (D-1)/2 errors. We adopted binary BCH ECC (i.e., Bose-Chaudhuri-Hochquenghem Code) for our scheme. BCH code is a multilevel, cyclic, error-correcting, variable-length digital code used to correct errors of up to approximately 25% of the total number of digits. Like other linear ECC codes, the first K bit digits in its N bit codeword are exactly the same as its original K bit message bits. This is the main motivation for us to select BCH code. In the procedure of content signing, the output (codeword) from the ECC encoder can still be separated into two parts: the original message and its corresponding parity check bits (PCB). We can then embed the PCB into the image as a watermark. In the procedure of content authentication, the authenticity can be verified by checking the syndrome of the merged codeword (message comes from the feature set extracted from the image to be verified and PCB comes from the watermark extracted from the same image).

D. Distortion-bounded watermark embedding and extraction under authentication bit-rate

Given LABR, from the EBCOT rate control mechanism we could quantitatively locate the truncation points for each code-block and estimate its resulted distortion as well. This advantage not only benefits the feature extraction but also brings us convenience in watermarking. For example, if the embedding process is to directly replace the selected bit plane p with the bits from the watermark to be embedded, the maximum distortion should be around the amount given in (1).

In our solution, the watermarks we embed are PCBs of ECC codewords. Since the system security is mainly guaranteed by crypto hashing on all concatenated ECC codewords (Messages + PCBs, *module A* in Fig. 5 and 6) and the generated digital signature, leaving watermarks (i.e., PCBs) public will not harm the system security. Considering the PCBs to be embedded are binary, we adopted some binary watermarking solutions in this case. To ensure that the embedded watermark can be correctly extracted under some incidental distortions, here we need to employ ECC again to re-encode PCB to generate the water-mark for embedding. We can embed the watermark in two ways as illustrated in Fig. 10. One way is to embed the watermark into a different location from where the watermark is generated. For instance, we can generate the signature from the HL subband and embed it into the LH subband at the same resolution level. The other way is to generate the signature in

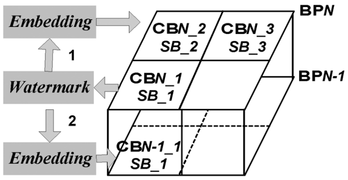


Fig. 10. Two ways of watermarking

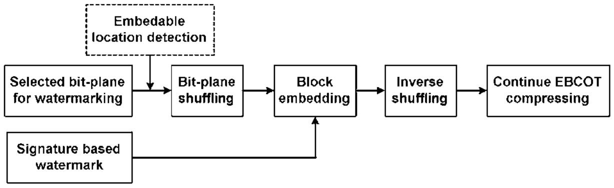

Embedable location detection
Selected bit-plane for watermarking
Bit-plane shuffling
Block embedding
Inverse shuffling
Continue EBCOT compressing
Signature based watermark

Fig. 11. Binary watermark embedding modified from [17]

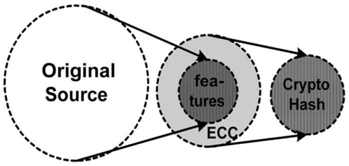


Fig. 12. Illustration of the security analysis

$a_1$bpp, which is a little bit less than LABR $a$bpp. This leaves the room between a and a_1 for watermarking. In our implementation, we adopt the first solution only for the sake of simplicity. As to the detailed watermark embedding, there are also two options. Once the watermark is formed and the watermarking location is selected, we embed the watermark into specified bit-planes under LABR. It means that the bit-planes to be watermarked are just above the marginal bit-planes to be cut for fitting a targeted CBR or a transcoded bit-rate. Note that for those codeblocks where the watermark cannot be embedded based on Human Vision System (HVS), we generate a secure random binary sequence for the features extracted from the corresponding codeblock for the final crypto hashing. In the case of high bit-rate compression, our watermarking strategy is to simply substitute watermark bits for the shuffled bit-plane bits and then perform inverse shuffling. In the case of low bit-rate compression, because higher bit-planes will be involved in watermarking and they usually contain more significant image information such as edges, we incorporate the module of detecting watermarkable coefficients to improve the visual quality of watermarked image, as shown with dashed-box in Fig. 11. For more details on detecting watermarkable coefficients, please refer to [17].

Watermark extraction is simply the inverse procedure of watermark embedding. After watermark embedding, the bit-planes above LABR will be re-coded by re-running EBCOT only in those portions, since watermarking has changed part of the contextual relationship among those bit-planes.

Based on the above description, we can see that the role of watermarking in our proposed scheme is only for storing the parity check information of ECC so that the errors of extracted features caused by incidental distortions can be corrected back in the authentication procedure.

IV. Performance Evaluation

A. Illustrative security analysis of proposed solution

Our analysis will only focus on the content hash function (i.e., feature extraction, ECC and crypto hashing) because it is the core of our proposed scheme for generating the signature and takes the responsibility for system security. We denote the performance of system security in terms of the possibility of the system being attacked, i.e., given an image and LABR, the possibility of finding another image which owns exactly the same content based hash value (i.e., it can pass the signature verification), under the same LABR.

We start by comparing our content-based hash function with traditional crypto hash functions such as 160 bits SHA-1 [1] whose security strength is about: $P_C \approx 2^{-80}$, under a birthday attack* assumption. It is usually deemed as nearly perfectly secure based on current computational capabilities. In our proposed scheme, in addition to the crypto hashing, the other two processing modules are feature extraction and ECC. Therefore, the security performance in our scheme comprises of three probabilities: P_F in feature extraction, P_E in ECC and P_C in crypto hashing which is already 2^{-80}. Since they are mutually independent and very small, the final system security performance could be drawn as:

$$P = 1-(1-P_F)(1-P_E)(1-P_C) \approx P_F + P_E + P_C\,.$$

Obviously P is larger than P_C. Therefore, we only need to study P_F and P_E here. In fact, P_F and P_E impair the system security in different ways, as shown in Fig. 12. A good feature descriptor should represent the original source as closely as possible. Differing from feature extraction, which functions as “removing” redundancy from original source, ECC functions as “adding” redundancy in order to tolerate incidental distortions. A good feature extraction method and a proper ECC scheme are the key factors in system security.

In order to analyze P_F, we make the following assumption:

Assumption 1: Lossy compression such as JPEG2000 is the approximate representation of its original source in terms of an optimal rate-distortion measure.

Intuitively, it implies that no security gap exists between the original source and its compressed version under a given targeted bit-rate. Therefore, we would argue that if a content hashing function could make use of all compressed information (e.g., all quantized coefficients in all blocks) at a targeted bit rate, it should be deemed the same as having made use of all of its original source. In other words, if an attacker intends to modify the content in the spaces between the original source and its compressed version, this attack should not be considered as harmful to this content (i.e., cause the meaning of content to change) because eventually the attacked content will be discarded by its lossy compression. In our scheme, we used almost all compressed information (e.g., the state of significant pass and the estimated distortions) above LABR for generating hash. Therefore we could argue that P_F is negligible (small enough) here under Assumption 1.

* A typical brute-force attack against crypto hash function: an adversary would like to find two random messages, M and M', such that $H(M) = H(M')$. It is named as birthday attack as it is analogous to randomly finding two people with the same birthday [1].

TABLE 1 (7,4) Hamming Code

Message	Codeword		Message	Codeword	
	Message	PCB		Message	PCB
0 0 0 0	0 0 0 0	0 0 0	1 0 0 0	1 0 0 0	0 1 1
0 0 0 1	0 0 0 1	1 1 1	1 0 0 1	1 0 0 1	1 0 0
0 0 1 0	0 0 1 0	1 1 0	1 0 1 0	1 0 1 0	1 0 1
0 0 1 1	0 0 1 1	0 0 1	1 0 1 1	1 0 1 1	0 1 0
0 1 0 0	0 1 0 0	1 0 1	1 1 0 0	1 1 0 0	1 1 0
0 1 0 1	0 1 0 1	0 1 0	1 1 0 1	1 1 0 1	0 0 1
0 1 1 0	0 1 1 0	0 1 1	1 1 1 0	1 1 1 0	0 0 0
0 1 1 1	0 1 1 1	1 0 0	1 1 1 1	1 1 1 1	1 1 1

To check P_E, originated from [15, 16], we have:

Lemma 1: Let H_C be our proposed content hash scheme based on an ECC scheme (N, K, D) with the error correction ability t (e.g., t = (D - 1) / 2). For any K' which satisfies $\|N - N'\| \leq t$, we have $H_C(N) = H_C(N')$. Note that in our JPEG2000 authentication scheme, K and K' are actually the extracted feature set, and N and N' are formed ECC codewords in the procedure of content signing and authentication, respectively. It means that upon properly selecting an ECC scheme, all corrupted codewords will be deemed authentic as long as they are still within the error correction capability. Therefore a proper ECC scheme could also make P_E negligible.

Clearly, ECC diminishes system security in some sense as ECC does provide the property of fuzziness. This goes back again to the issue of how to select a proper ECC scheme to balance between system robustness and security. However, we have to accept the fact that ECC does introduce some security risk into the system. Refer to Table 1, for instance, if the formed codeword is 0000000, 7 other codes will be considered as acceptable in the verification: 1000000, 0100000, 0010000, 0001000, 0000100, 0000010 and 0000001, because they are all within the designed error correction capability (correcting 1 bit error) and will be corrected as 0000000 in the procedure of signature verification. Now, the question is, is accepting these 7 codes secure? It urges that an application oriented ECC scheme under pre-defined acceptable manipulations plays a key role in semi-fragile content-based authentication.

It's also interesting and important to understand the interplay between decisions based on the block-based syndrome verification and the global verification. The local check is based on syndrome calculation of the formed codeword where the feature computed from the received image and the PCBs recovered from the watermark that is generated and embedded at the source site. If the formed codeword is not correctable, we can claim a failure of authentication based on such block-based verification. However, since we do not transmit the entire codeword, there exist changes of a block that cannot be detected (as the case 0001111 vs. 1111111 which own the same PCBs, syndrome calculation will say ok). Instead, such changes will be detected by the global crypto hash value, because the hash is generated using the entire codewords, not just the PCBs. Therefore, the following cases may happen in our solution: the image is deemed unauthentic because of inconsistency between hashed sets while we are unable to indicate the locations of attacks because there are no uncorrectable codewords found. In such cases, we still claim the image is unauthentic although we are not able to indicate the exact alternations.

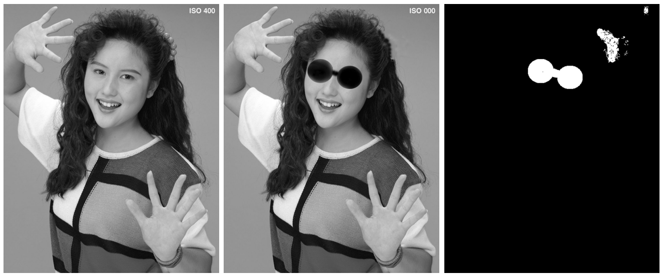


Fig. 13. Examples of original watermarked image (left), attacked watermarked image (middle) and their differences (right)

The last concern related to security is watermarking. In our proposed scheme we do not need to pay more attention to watermark security since watermarking here only functions to store part of ECC information.

B. Experimental methodology and results

To evaluate system performance on robustness, two important measures used are false acceptance rate (FAR) and false rejection rate (FRR). The FRR is the percentage of failed attempts in generating its derived signature from the authentic image, averaged over all authentic images in a population O. The FAR is the percentage of success among attempts in generating its derived signature from the forged image, averaged over all forged images in a population M. Faced with some practical difficulties when we conduct testing such as manually forging a number of meaningful images, we measure FAR and FRR in an alternative way. Our FAR and FRR are derived based on codeblock-based evaluation, not image-based evaluation. Given a pair of images to be tested (One is the original watermarked and the other one is its fake version), we know the exact codeblock-based modification information in terms of their wavelet transform representation. Assume that there are a total of N codeblocks in all subbands in the pair of images, M blocks are modified and O blocks remain unchanged. N = M + O. We then pass the pair of these images (e.g., watermarked image and modified watermarked image) through various kinds of acceptable manipulations such as multi-cycle compression, transcoding, etc. FAR and FRR will be derived based on checking all these distorted images codeblock by codeblock. Assuming that after distorting images, M' among M blocks can pass the authentication and O' among O cannot pass the authentication respectively, FAR and FRR could be obtained as $FRR = O'/O$ and $FAR = M'/M$.

Since JPEG2000 has very rich flexibilities in code stream syntax as well as the coding schemes, we fix some settings such as tiling and skip some JPEG2000 modules such as color transformation in order to focus more on illustrating the validity of our proposed solution. Unless the parameters and settings are particularly specified, they will remain their

TABLE 2 Default Parameter Setting For System Tests

Parameters selected for JPEG2000 and ECC	Defaults
Progressive model	SNR with 50 layers
Codeblock	64x64
Sub-codeblock size	16x16
Quantization step size	Implicit, 0.007825 (normalized)
WT filter	9x7 floating
Decomposition levels	5
ECC schemes (signature generation)	BCH(255, 239/215/179, 2/5/10):Pass features BCH(7, 4, 1) / (15, 7, 2): Distortion features
ECC schemes (watermark embedding)	BCH (31, 16, 3) and (15, 7, 2)
Watermarking	Feature extracted from LH and embedded into HL and HH. Method: [22]
Others	Input: 256 gray-level image, no tiling,

defaults as shown in Table 2.

We conducted testing on 3 pairs of images. The size of all images has been cropped to 512x640. Fig. 13 shows one of three pairs of images. At full bit-rate, the total modifications of 3 pairs of images are summarized in Table 3. In our testing, we take them as the ground-truth for evaluation in terms of FAR and FRR. We further divide each codeblock (64x64) into sub-codeblock whose size is 16x16 for locating potential modifications more accurately. The feature extraction and watermarking will be based on the sub-codeblock.

1). Tests under multiple compression cycles with full bit-rate, intermediate bit-rate and low bit-rate

We coded and watermarked images with JPEG2000 under different target CBRs and LABRs. The PSNR between coded JPEG2000 images and coded plus watermarked JPEG2000 images is shown in Fig. 14(a) under different bit-rates. Both watermarked and modified watermarked images pass the recompression procedure (decoding-and-encoding) with 1, 5, and 10 cycles at the CBRs. We select 3 bit-rates for our tests: full bit-rate (its actual CBR is around 2.5-4bpp and LABR: 2bpp), 1bpp (LABR: 0.8bpp) and 0.25bpp (LABR: 0.2bpp). We also tested 3 different ECC schemes: ECC1 (255, 239, 2), ECC2 (255, 215, 5) and ECC3 (255, 179, 10) because we observed that the corrupted distortion is more severe in low bit-rate than in high bit-rate, especially in the case where the tested bit-rate is obtained through different transcoding ways such as directly encoding, parsing and truncation. The average results are shown in Table 4. Readers may notice that the LABR is slightly less than the CBR in our tests. The main reason is that we use LABR to control both signature generation/verification and watermark embedding/extracting. To ensure a watermark is extracted correctly under multiple compression cycles, ECC is not enough to tackle the incidental distortions if we watermark on some marginal bit-planes. Here we assume our acceptable image manipulation is multi-cycle JPEG2000 compression.

TABLE 3 Number of Attacked Blocks in Full Bit-Rates

ResLev	0		1		2		3		4		5	
	O	*M*	*O*	*M*	*O*	*M*	*O*	*M*	*O*	*M*	*O*	*M*
Image 1	3	1	9	3	30	6	105	15	453	27	1791	129
Image 2	3	1	9	3	27	9	108	12	459	21	1854	66
Image 3	2	2	6	6	21	15	90	30	417	63	1713	207

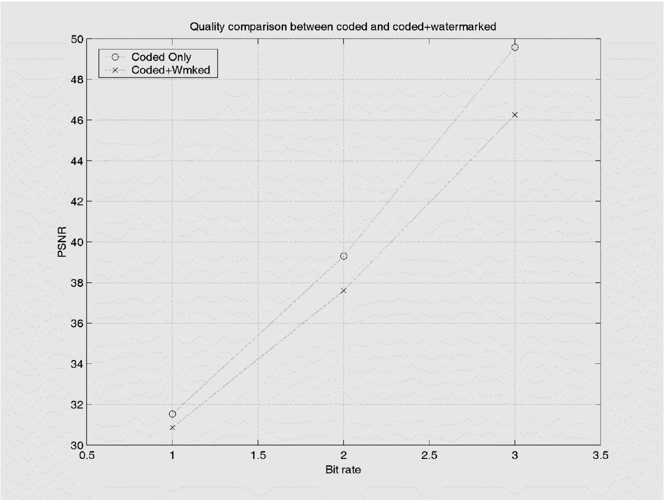


Fig. 14(a) PSNR comparison on coded and code+watermarked images

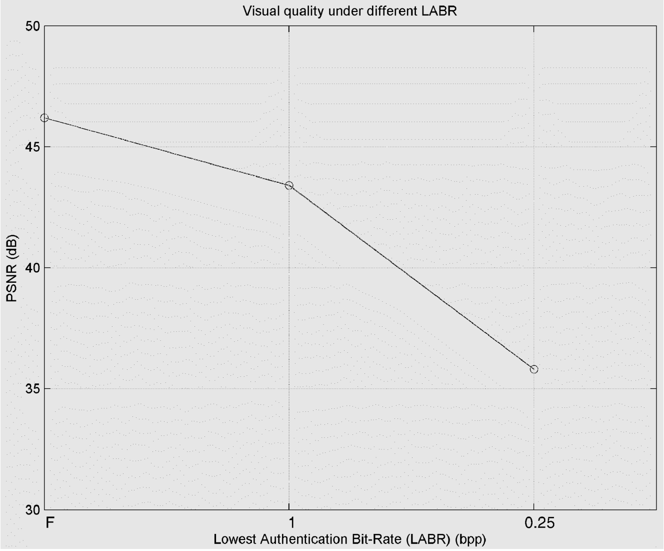


Fig. 14(b) PSNR comparison on different LABRs

Fig. 14(b) shows the results of the visual impacts on the distance between CBR and LABR in terms of PSNR. We set the CBR to full bit-rate while LABR is set to 0.25, 1bpp and full bit-rate (2bpp) respectively. With the LABR decreasing from 2bpp to 0.25bpp, the PSNR is also decreasing from 46dB to 36dB. The farther the distance between targeted CBR and LABR, the worse the image quality we receive after watermarking. This is because for a lower LABR we have to embed the watermark into a higher bit-plane which affects the image quality more effectively. Remember that a lower LABR also requires the feature to be extracted from a higher bit-planes, so a lower LABR also results in a more robust authentication.

TABLE 4 Testing Results Under Multi-cycle compressions

Results	Full bit-rate		Intermediate bit-rate		Low bit-rate	
	FAR	**FRR**	**FAR**	**FRR**	**FAR**	**FRR**
Image 1	ECC1: 0.00 ECC2: 0.01 ECC3: 0.01	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.00 ECC2: 0.01 ECC3: 0.02	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.00 ECC2: 0.00 ECC3: 0.01	ECC1: 0.06 ECC2: 0.05 ECC3: 0.00
Image 2	ECC1: 0.00 ECC2: 0.00 ECC3: 0.01	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.00 ECC2: 0.02 ECC3: 0.02	ECC1: 0.01 ECC2: 0.00 ECC3: 0.00	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.07 ECC2: 0.05 ECC3: 0.00
Image 3	ECC1: 0.01 ECC2: 0.02 ECC3: 0.02	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.01 ECC2: 0.03 ECC3: 0.03	ECC1: 0.00 ECC2: 0.00 ECC3: 0.00	ECC1: 0.00 ECC2: 0.00 ECC3: 0.01	ECC1: 0.06 ECC2: 0.05 ECC3: 0.00

Actually the authentication robustness is not only related to

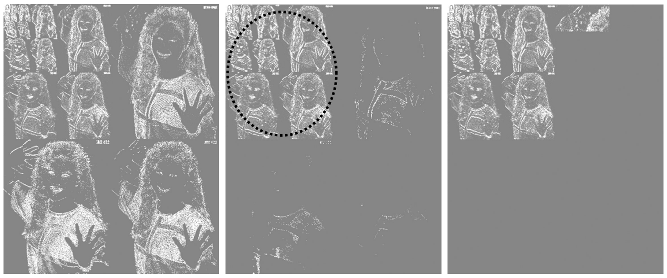


Fig. 15. Illustration on how transcoding and progression orders affect the authentication. (a) Image with 2bpp (Res progressive and SNR progressive) (b) Image with 0.8bpp parsed or truncated with SNR progressive and parsed with resolution progressive (c) Image with 0.8bpp truncated with resolution progressive. Therefore we can only protect the labeled part if no information about transcoding and progression order is available.

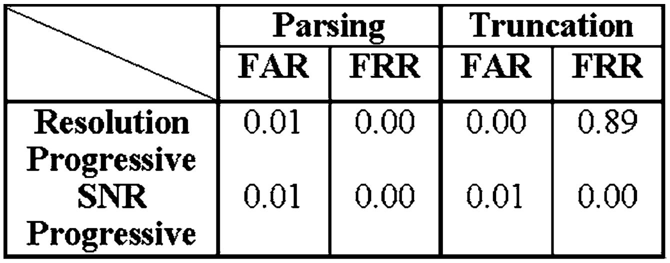
LABR setting, but also related to ECC scheme. The stronger the ECC error correction capability is, the more robust the system will be. Though in the examples in Table 4, the empirical results show that these 3 ECC schemes work well for 3 different bit-rate ranges; however, stronger ECC will also increase FAR which is another important measure of system.

Note that Table 4 is on the basis of codeblock, not whole image. The final authentication results should be decided on its global level. For instance, if we adopt ECC1 for low bit-rate cases, almost all authentic images will be rejected as attacked images. If we adopt ECC3 for full bit-rate cases, we still can differentiate the attacked images from authentic images based on comparing two crypto hash values, but some attacked code-blocks will not be detected (FAR is increasing).

2). Tests under different transcoding and progression order

JPEG2000 provides two ways of transcoding: parsing and truncation. In this test, we show how the different progression order and transcoding affect the performance of our proposed solution. We compress the image at 2bpp (LABR: 0.6bpp) in two progression orders: resolution-based and SNR-based, respectively, while our signature generation/watermarking remains SNR-based. Then we transcode them to 0.8bpp through parsing and truncation. Table 5 summarizes FAR and FRR by adopting ECC2. We can see that different progression order and transcoding affect the authentication results a lot. If the final bit-stream is in SNR progression order, then the authentication works well in both parsing and truncation transcoding. However, if the final bit-stream is in resolution progression order, parsing-based transcoding still works stably while truncation-based transcoding may not work anymore. The inconsistence between the progression order of image signing (e.g., SNR progression) and that of final compression (e.g., resolution progression) means some codeblocks which are part of the feature set under LABR will be cut by transcoding. Fig. 15 shows an image with different transcoding methods under different progression orders. Therefore, knowing progression order and possible transcoding before signing and verifying will be helpful in accurately and stably authenticating JPEG2000. In practice, such information can be found from the file header of a JPEG2000 image, which is also necessary for correctly decoding JPEG2000 itself. If we cannot extract such information, we can only perform coarse content integrity protection. For instance, we only protect the intersectional part of image among different progression orders, as shown in Fig. 16.

TABLE 5 Testing Results Under Different Transcoding



	Parsing		Truncation	
	FAR	FRR	FAR	FRR
Resolution Progressive	0.01	0.00	0.00	0.89
SNR Progressive	0.01	0.00	0.01	0.00

3). Testing on compatibility with ROI

Encoding Region-of-interest (ROI) is a new feature in the JPEG2000 standard [18]. Refer to Fig. 1, where the module of ROI encoding is actually between quantization and EBCOT. Therefore, our proposed scheme also works well with ROI coding. Refer to Fig. 16, we define one rectangular ROI at (left: 150, top: 240, width: 350, height: 400) and encode with MAXSHIFT method [18], see Fig. 16(a). Its target CBR is around 2.7bpp while the LABR is set to 0.6bpp. Fig. 16(a) shows the decoded image with watermarking ROI model in 0.6bpp. The authentication result is shown in Fig. 16(b). As shown in Fig. 16(b), the image can pass the authentication because the whole ROI is protected at 0.6bpp. In comparison with the no-ROI case, Fig. 16(c) shows the decoded image with watermarking non-ROI model in 0.6bpp. Its corresponding authentication result is shown in Fig. 16(d).

V. Conclusion and Future Work

In this paper, we have proposed an integrated content-based authentication scheme targeting at verifying the authenticity of JPEG2000 images quantitatively and securely in terms of a unique concept named lowest authenticable bit-rates (LABR). The main contributions of our solution are summarized below:

- JPEG2000 images can be quantitatively authenticated in terms of LABR: The proposed solution can quantitatively authenticate images specified by LABR. It will bring users more convenience and achieve a good trade-off among system robustness, security and image quality.
- Error Correction Coding (ECC) is employed to tame the incidental distortions and to obtain only one crypto hash / signature per image regardless of its size, in a semi-fragile way: ECC's message comes from the extracted features of the image. ECC's parity-check-bits (PCB) are the seeds of watermarks and will be embedded back into the image. Such novel usage of ECC will bring the system the ability to correct some possible errors in its extracted features without increasing system payload: The codeblock based watermarks (PCBs) could be used for locating malicious modifications and the final content based signature is obtained by cryptographically hashing all corresponding codewords (message + PCB) to make sure no catastrophic security holes exist.

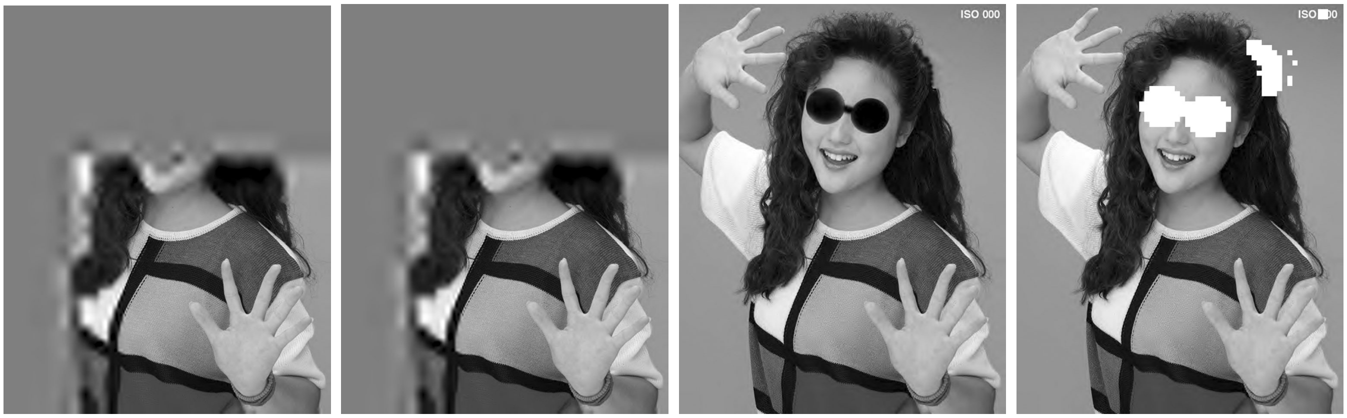


Fig. 16. Authenticating ROI. (a) Decoded with ROI image: 0.6bpp (b) Authentication result: 0.6bpp with ROI (LABR: 0.5bpp). Since no change within ROI, the testing image still passes the verification. (c) Decoded without ROI: 0.6bpp (d) Authentication result: 0.6bpp without ROI (LABR: 0.5bpp).

- Flexibly to be incorporated into various security protocols (Symmetric / Asymmetric): Working under PKI will also make the proposed solution be easily incorporated into current data-based (i.e., fragile) authentication platforms.
- Fully compatible with JPEG2000 coder (Part 1): All features are directly from EBCOT and the generated watermarks are also embedded back in the procedure of EBCOT. The proposed solution could efficiently co-work with the procedure of JPEG2000 image encoding and decoding. The authentication only needs one mandatory parameter: LABR, which is similar to CBR.

Actually LABR setting affects system security, system robustness and image quality. Intuitively a smaller LABR will result in a more robust authentication, a lower image quality and a less secure image because a smaller LABR means to extract the features from the coarser image content (i.e., a lower resolution, a higher bit-planes) and also means to embed the watermarks into the coarser image content. Usually the coarser image content will be more robust to the bounded distortions because the noise between LABR and CBR will not affect the system robustness.

Though in this paper we have given some guidelines on LABR setting and ECC scheme selection, how to quantitatively set the LABR and select a suitable ECC scheme is still an issue under study for a specific application. Defining the acceptable image manipulations (e.g., JPEG compression) may not be enough to set a proper LABR. Their strengths are also required (e.g., the Quality Factor defined in JPEG compression). Similarly, ECC setting also affects both system robustness and system security. The system robustness set by LABR could also be compensated by increasing the error correction capability of the ECC scheme (Refer to Fig.12). It is worth noting that LABR setting and ECC scheme selection are also closely related to FAR and FRR which are two important system performance measures for real applications.

Currently we are studying the distortion bounds of some common acceptable manipulations such as multi-cycle JPEG /JPEG2000 compressions. We aim to use it for optimizing the interplay between LABR setting and ECC scheme selection, in a theoretic way.

Future work is to study more robust solutions under the proposed framework to tolerate other acceptable manipulations such as image filtering/sharpening, contrast and intensity change, etc. Other research issues may include studying good feature representation for other media (audio/video) under other formats (e.g., MPEG1/2/4).

Acknowledgment

The authors would like to thank Mr. K. Maeno (Oki Electric Industry Co., Ltd.) and Dr. C.-Y. Lin (IBM) for very helpful discussions. The source of testing images: www.jpeg.org.

References

[1] B. Schneier, Applied Cryptography, New York: Wiley, 1996

[2] P. W. Wong and N. Memon, Secret and public image watermarking schemes for image authentication and ownership verification, IEEE Transactions on Image Processing, Vol.10, No.10, pp.1593-1601, 2001.

[3] P.-C. Su, H.-J. M. Wang and C.-C. J. Kuo, An integrated approach to image watermarking and JPEG2000 compression, Journal of VLSI Signal Procesisng, No.27, pp.35-53, 2001.

[4] M. Schneider and S.-F. Chang, A robust content-based digital signature for image authentication, ICIP'96, 1996.

[5] C.-Y. Lin and S.-F. Chang, A Robust Image Authentication Method Distinguishing JPEG Compression from Malicious Manipulation, IEEE Transactions on Circuits and Systems for Video Technology, Vol.11, No.2, pp.153-168, Feb. 2001

[6] C.-S. Lu and H.-Y. Mark Liao, Structural Digital Signature for Image Authentication: An Incidental Distortion Resistant Scheme", to appear in IEEE Trans. on Multimedia, 2002.

[7] D. Johnson and A. Menezes, The Elliptic Curve Digital Signature Algorithm (ECDSA), 1999, in http://cacr.math.waterloo.ca;

[8] J. Fridrich and M. Goljan, Robust hash functions for digital watermarking, Proceedings of IEEE International Conference on Information Technology - Coding and Computing'00, Las Vegas, March, 2000.

[9] L. Xie, G. R. Arce and R. F. Graveman, Approximate image message authentication codes, IEEE Trans Multimedia, Vol.3, No.2, pp.242-252, 2001.

[10] R. Venkatesan, S.-M. Koon, M. H. Jakubowski and P. Moulin, Robust image hashing, Proceedings of IEEE International Conference on Image Processing'00, Vancouver, Canada, Sept, 2000.

[11] Information Technology-JPEG2000 Image Coding System, ISO/IEC International Standard 15444-1, 2000.

[12] JJ2000: An implementation of JPEG2000 in JAVATM, available at http://jj2000.epfl.ch.

[13] D. Taubman, High performance scalable image compression with EBCOT, IEEE Transactions on Image Processing, Vol.9, No.7, pp.1158-1170, Jul. 2000.

[14] D. Taubman, Report on core experiment CodeEff22 (EBCOT: Embedded Block Coding with Optimized Truncation), ISO/IEC JTC1/SC29/WG1N1020R, Oct. 1998.

[15] G. I. Davida, Y. Frankel and B. J. Matt, On enabling secure applications through off-line biometric identification, Proceedings of the 1998 IEEE Symposium on Security and Privacy, pp.148-157, 1998.

[16] S. Lin and D. J. Costello, JR., Error control coding: Fundamentals and applications, Prentice-Hall, 1983.

[17] M. Wu, E. Tang and Bede Liu, Data hiding in digital binary image, Proceedings of ICME'00, 2000.

[18] C. Christopoulos J. Askelof and M. Larsson, Efficient methods for encoding regions of interest in the upcoming JPEG2000 still image coding standard, IEEE Signal Processing Letters, Vol.7, No.9, pp.247-249, 2000.

Author (M’xx–SM’xx–F’xx) and the other authors may include biographies at the end of regular papers.