Packet-Level Naïve JPEG2000 Information Hiding Using Redundant Packet Insertion Technique

Chiang-Lung Liu, Chien-Chung Lee, and Shih-Wei Lin
Department of Electrical and Electronic Engineering
Chung Cheng Institute of Technology
National Defense University
Tahsi, Taoyuan 33509, Taiwan, ROC
E-mail: chianglung.liu@gmail.com

Abstract*-Naïve information hiding embeds information data in cover carrier for practical applications such as error correction or image authentication. The positions the information data resides in are not necessarily kept secret. Direct insertion of information data into JPEG2000 codestream can effectively reduce the processing time of information embedding and extraction. However, it may also result in decoding failure for a standard JPEG2000 image viewer. In this paper, we propose a packet-level naïve JPEG2000 information hiding method that uses a redundant packet insertion technique to solve these problems simultaneously. Experimental results show that the full or truncated information-embedded codestream can be correctly decoded by several JPEG2000 Part-1 compliant decoders. That is, the proposed method can meet both the syntax compliance and scalability requirements. Moreover, because the proposed information embedding and extraction processes are performed in packet level, very few decoding operations are involved in the proposed method. That is, the proposed method can meet the simplicity requirement. Therefore, the proposed naïve JPEG2000 information hiding method is very effective and practical for various JPEG2000 applications.*

Keywords: JPEG2000, naïve information hiding, redundant packet insertion, syntax compliant, scalable.

1. Introduction

Information hiding is a technique that can hide secret information into cover carriers for various applications. There have been different kinds of information hiding methods [1] proposed in the literature. Among them, digital images may be the most popular cover carriers because they can be easily accessed on the Internet. JPEG2000 [2-4] is the latest standard for still image compression and provides excellent performance under low-bit-rate compression. Therefore, JPEG2000 is fast becoming the solution of choice for modern multimedia applications. According to the types of application, JPEG2000 information hiding techniques can be broadly divided into two categories: steganography [5-12] and digital watermarking [13-16].

Steganography is referred to as covered writing that embeds secret information in innocuous-looking cover carriers for covert communication. In this type of application, the positions of the cover carrier the secret information resides in should be only shared by the communication parties. For robust digital watermarking, the information (called watermark) is embedded in the protected image to be a proof of ownership. The embedded watermark should be robust enough to resist various kinds of image processing operations. On the other hand, fragile watermarks are embedded to detect any modifications to the protect images.

Recently, the information hiding concept has been adopted by several JPEG2000 related applications to convey the required information between the sender and receiver. In these applications, the positions the information data resides in are not necessarily kept secret. For example, error correction codes [17] can be embedded in the JPEG2000 frame to recover the corrupted data resulted from the transmission error. For the application of image authentication, the encrypted hash code can be embedded into JPEG2000 codestream for verifying the integrity of the protected image [18]. We call this kind of applications the *naïve information hiding*. Intuitively, most of the steganographic methods can be used for naïve information hiding. However, because the methods used for secret

communication aim at hiding the existence of secret information, the payload provided by these methods is quite limited. Moreover, to avoid being detected by steganalytical detectors, most of JPEG2000 steganography embeds the secret information in discrete wavelet transform (DWT) domain. It means that the information extractor should reverse most of the encoding processes to retrieve the embedded information. It may not be practical for naïve information hiding of time demand such as error correction of video stream. One of the solutions to solve this problem is to directly embed the information data into the JPEG2000 codesteam. However, direct insertion of information data in a JPEG2000 codestream may break the format of the codestream and result in decoding failure for standard JPEG2000 image viewer. It means that the information data embedded JPEG2000 codestream should be still decoded by JPEG2000 Part-1 compliant decoder. In this paper, this is referred to as *syntax compliance requirement*. In practical applications, JPEG2000 codestreams may be truncated to meet lower-bit-rate requirement. That is, the information data embedded in JPEG2000 codestream should survive the truncation operation. This is an important JPEG2000 feature and is referred to as *scalability requirement* in this paper. According to the description provided above, we conclude that a naïve JPEG2000 information hiding method should possess the following properties:

(1) Syntax compliance: the information embedded codestream should be correctly decoded by any JPEG2000 Part-1 compliant decoder without causing decoding failure.
(2) Scalability: the information embedded codestream can be directly truncated for lower-bit-rate applications.
(3) Simplicity: the information hiding method should be simple and effective so that the embedded information can be easily accessed by the communication party.

In this paper, we propose a naïve JPEG2000 information hiding method to meet these requirements simultaneously. The proposed method uses a *redundant packet insertion* (RPI) technique to elegantly insert information data into the JPEG2000 codestream. More specifically, the information data are packed as several redundant packets which are further embedded right behind the selected packet(s). Because these redundant packets will be skipped in the decoding process, the information embedded codestream can be correctly decoded by a JPEG2000 Part-1 compliant decoder. That is, the proposed method can meet the syntax compliance requirement. It should be noted that, to our knowledge, the proposed RPI technique is the first packet-level information hiding technique that can achieve this purpose. To meet the scalability requirement, the redundant packets can be inserted in the significant part of the codestream to avoid being truncated even in a very low-bit-rate application. In Section 3, we will show that the proposed RPI technique is simple enough to meet the simplicity requirement.

To completely describe the proposed naïve information hiding method, the rest of this paper is organized as follows. In Section 2, we briefly introduce the JPEG2000 coding scheme. The internal structure of JPEG2000 codestream is then concisely described. In Section 3, the proposed information embedding and extraction processes is first detailed, followed by an example of the proposed RPI technique. Several experimental results are demonstrated in Section 4 to show the effectiveness of the proposed method. Section 5 concludes this work.

2. Overview of the JPEG2000 coding scheme

JPEG2000 refers to all parts of the standard. Part 1 (the core coding system) is now published as an International Standard. Parts 2 - 12, except Part 7, are complete or nearly complete. In this paper, we focus on the introduction of the first part of the standard. The overview of the JPEG2000 coding scheme in this section is restricted to application of the proposed RPI technique, especially the format of JPEG2000 codestream. Interested readers may refer to [2-4] for details of JPEG2000.

JPEG2000 is a wavelet-based image coding standard. In JPEG2000, an image can be partitioned into smaller rectangular regions called tiles. Each tile is encoded independently as through they were entirely distinct images and can be divided into several color components. Each tile-component can be further decomposed into different resolution-levels using a 2-dimension DWT. Applying the DWT continuously on each lowest frequency sub-band (usually referred to as LL sub-band) generates a series of sub-bands belonging to different transform levels.

After wavelet decomposition, each sub-band is divided into rectangular blocks called precincts. Each precinct is further divided into smaller blocks called code-blocks. Each code-block is individually quantized and entropy-encoded to generate a bit-stream which consists of a number of bit-plane coding passes. All the data assembled during a coding pass in a tile forms a layer which

represents an image quality increment. Within the precinct, all the spatially consistent code-blocks are grouped together into a packet.

A JPEG2000 codestream is composed of a main header and a set of packets. Figure 1 shows the structure of a JPEG2000 codestream. The main header stores the information for decoding the codestream. Packets are the fundamental building blocks in JPEG2000 codestream. A packet comprises a packet header and the compressed bit-stream from code-blocks belonging to a specific component, resolution level, precinct, and layer. The order in which packets appear in the codestream is called the progression order. JPEG2000 supports progression in four dimensions: layer (L), resolution level (R), precinct (P), and component (C). That is, the packets in the codestream are arranged according to the selected progression order so that the image quality can be constructed in the same order. JPEG2000 supports five progression orders: LRCP, RLCP, RPCL, PCRL, and CPRL.

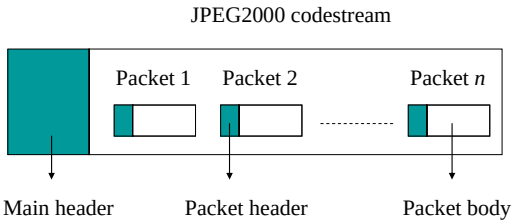

Figure 1. Structure of JPEG2000 codestream

3. The proposed method

The proposed naïve information hiding method is performed in the packet level and comprises information embedding and extraction stages.

3.1. Information embedding

The proposed naïve information hiding method embeds information data right behind the selected packet(s) in JPEG2000 codestream. However, direct insertion of the information data in a JPEG2000 codestream may break the format of the codestream and result in decoding failure. The proposed method uses the redundant packet insertion (RPI) technique to solve this problem. In this section, we first introduce the concept of the RPI. Detailed description of the proposed information embedding process is then followed.

JPEG2000 uses the first bit of the packet header to indicate whether the packet contains data or not. If the first bit of the packet header is 1, the packet is non-empty and the other bits of the packet represent the real data and will be processed in the decoding process. On the other hand, if the first bit of the packet header is 0, the packet is an empty packet. In this case, the packet is considered as a one-byte packet and will be skipped in the decoding stage. In this paper, this kind of packets is specially referred to as zero packets. The proposed RPI technique uses zero packets to embed the information data. More specifically, the information data is first partitioned into 7-bit information blocks. Each information block is then stuffed into the 7 least significant bits (LSBs) of a zero packet. In this paper, we call the zero packets stuffed with information bits the information packets, and the zero packets stuffed with 0's the empty packets.

For JPEG2000 codestream, a packet can be identified by four factors: component (C), precinct (P), resolution level (R), and layer (L). Once a JPEG2000 codestream is generated, only the layer factor can be changed without causing a decoding failure. The proposed RPI technique takes advantages of this property to insert information packets and some necessary empty packets right behind the selected packet(s) as if the codestream has originally been encoded with such many layers.

Figure 2 shows the concept of the proposed information embedding process. The detailed steps of the proposed information insertion process are as follows:

Step 1. Select n packets from the codestream to be attached with information packets.

Step 2. Let N_M denote the number of bits of the information data. Partition the information data into N_I 7-bit information blocks, i.e.,
$$N_I = \lceil N_M / 7 \rceil. \quad (1)$$

Step 3. Divide N_I information blocks into n information sections and calculate the number of information blocks, N_S, for each information section according to the following formula:
$$N_S = \lceil N_I / n \rceil, \quad (2)$$
where $\lceil x \rceil$ rounds x to the nearest integer towards infinity.

Step 4. Create $n \times N_S$ zero packets. Replace the 7 LSBs of the first N_I zero packets with the corresponding information block and leave the 7 LSBs of the other zero packets empty. That is, we create n information groups each with N_S information packets.

Step 5. Determine the progression order of the original codestream by decoding the main header. If the progression order is LRCP, Step 6 is performed. Otherwise, if the

progression order is RLCP, Step 7 is performed. Otherwise, Step 8 is performed.

Step 6. (Redundant packet insertion for progression order LRCP)

Step 6.1. Perform the following calculations:

$$q = \lceil N_S / (N_R \times N_C \times N_P) \rceil, \quad (3)$$

$$r = \mathrm{MOD}(N_S, N_R \times N_C \times N_P), \quad (4)$$

where N_R, N_C, and N_P denote respectively the number of resolution levels, the number of components, and the number of precincts, and MOD(x,y) operation takes the remainder of x/y.

Step 6.2. If r is 0, attach N_S information packets of ith information group to the end of ith selected packet and N_S empty packets to the end of each unselected packet, where $1 \le i \le n$. Otherwise, attach N_S information packets of ith information group and $N_R \times N_C \times N_P - r$ empty packets to the end of ith selected packet and $q \times N_R \times N_C \times N_P$ empty packets to the end of each unselected packet, where $1 \le i \le n$.

Step 6.3. Modify the number of layer in the main header with

$$N_L' = N_L + N_L \times N_R \times N_C \times N_P \times q, \quad (4)$$

where N_L and N_L' denote the original and the modified number of layer, respectively.

Step 7. (Redundant packet insertion for progression order RLCP)

Step 7.1. Perform the following calculations:

$$q = \lceil N_S / (N_C \times N_P) \rceil, \quad (5)$$

$$r = \mathrm{MOD}(N_S, N_C \times N_P), \quad (6)$$

where N_C and N_P denote respectively the number of components and the number of precincts, and MOD(x,y) operation takes the remainder of x/y.

Step 7.2. If r is 0, attach N_S information packets of ith information group to the end of ith selected packet and N_S empty packets to the end of each unselected packet, where $1 \le i \le n$. Otherwise, attach N_S information packets of ith information group and $N_C \times N_P - r$ empty packets to the end of ith selected packet and $q \times N_C \times N_P$ empty packets to the end of each unselected packet, where $1 \le i \le n$.

Step 7.3. Modify the number of layer in the main header with

$$N_L' = N_L + N_L \times N_C \times N_P \times q, \quad (7)$$

where N_L and N_L' denote the original and the modified number of layer, respectively.

Step 8. (Redundant packet insertion for progression orders RPCL, PCRL, and CPRL)

Step 8.1. Attach N_S information packets of ith information group to the end of ith selected packet and N_S empty packets to the end of each unselected packet, where $1 \le i \le n$.

Step 8.2. Modify the number of layer in the main header with

$$N_L' = N_L + N_L \times N_S, \quad (8)$$

where N_L and N_L' denote the original and the modified number of layer, respectively.

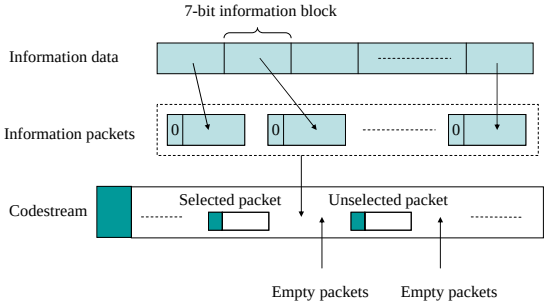

Figure 2. Concept of the proposed information embedding process

3.2. Information extraction

The way to extract the information data is very simple and straightforward. Figure 3 shows the concept of the proposed information extraction process. The information packets are first extracted from the back of the selected packet(s). Each 7-bit information block is then extracted from the corresponding information packet and attached to the former information block to form the final information data.

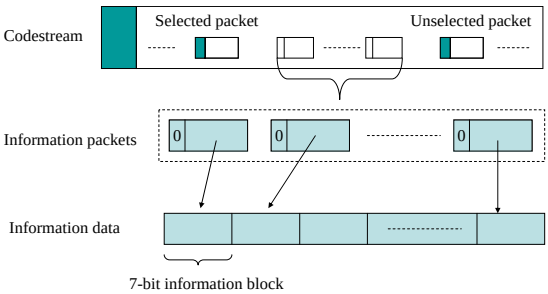

Figure 3. Concept of the proposed information extraction process

3.3. Example of redundant packet insertion

This section presents an example of the proposed redundant packet insertion technique

using various progression orders. Assume that the information data is of 1024 bits and the first two packets are selected to be attached with the information data. Therefore, the information data can be partitioned into $\lceil 1024/7 \rceil = 147$ information blocks which can be further portioned into 2 information sections each with $\lceil 147/2 \rceil = 74$ information blocks. Therefore, there are totally $74 \times 2 = 148$ information blocks that have been generated to be stuffed into 148 zero packets to form 148 information packets. The first to 74th information packets belongs to first information group and the 75th to last information packets belongs to the second group.

Assume that an image is encoded with 3 layers (i.e., $N_L=3$), 2 resolution levels (i.e., $N_R=2$), 1 component (i.e., $N_C=1$), and 2 precincts (i.e., $N_P=2$). For progression order LRCP, both the 74 information packets can be attached to the end of the first two packets. The insertion of information packets will virtually add $\lceil 74/(2\times1\times2) \rceil = 19$ layers to the original codestream. Because $\mathrm{MOD}(74, 2\times1\times2) = 2$ is not 0, additional $2\times1\times2-2=2$ empty packets should be attached to the end of information packets to make up the required number of packets of 19 layers. To maintain the synchronization of the packets, there are totally $19\times(2\times1\times2)=76$ empty packets that should be attached to the end of each unselected packet. Moreover, the number of layer recorded in the main header should be changed to $3+3\times2\times1\times2\times19=231$. Figure 4 shows the structure of the original codestream and the information-embedded codestream.

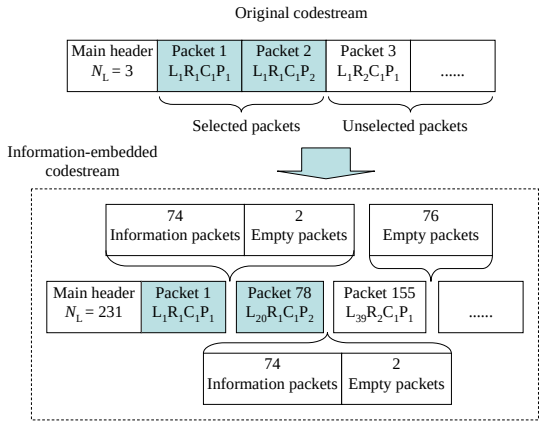

Figure 4. Example of the proposed redundant packet insertion for progression order LRCP

4. Experimental results

Various experiments have been performed to demonstrate the effectiveness of the proposed naïve information hiding method. A standard 256×256 gray image, Lena, was taken as the original image and was divided into 1 tile and 1 component. Each component was decomposed into 3-level DWT coefficients through our experiments. The DWT coefficients were divided into 64×64 code-blocks which was then encoded to form a JPEG2000 codestream. We also randomly generated a 1024-bit message as the information data.

4.1. Syntax compliance test

To show the proposed RPI technique can meet the syntax compliance requirement, we first generated five standard JPEG2000 codestreams for the original image using five different progression orders respectively. The information data was then inserted into each codestream using the proposed RPI technique. It should be noted that there were 4 selected packets in each codestream used to be attached with information packets. Several JPEG2000 compliant decoders, including IrfanView 3.91 [19], JJ2000 [20], PhotoImpact 10 [21], and ACDSee 7.0 [22], were used to decode the information-embedded codestreams. Experimental results show that all the information-embedded codestreams can be successfully decoded by these image viewers. It means that the proposed RPI technique can meet the syntax compliance requirement.

4.2. Scalability test

To show the proposed RPI technique can meet the scalability requirement, a JPEG2000 codestream was generated with the progression order LRCP and then inserted with information data using the proposed PRI technique. The information-embedded codestream was then truncated with various lower bit-rates and then decoded using different JPEG2000 Part-1 compliant decoders. Experimental results show that all the truncated codestreams can be successfully decoded. That is, the proposed RPI technique can meet the scalability requirement.

5. Conclusions

In this paper, we propose a packet-level naïve JPEG2000 information hiding method which allows the information data to be directly inserted into a JPEG2000 codestream. The information data is first partitioned into 7-bit information blocks and then stuffed into several zero packets to form information packets. An RPI technique is also proposed to elegantly attach the information

packets and some empty packets to the end of the selected packet(s).

Experimental results show that all the information-embedded codestream can be correctly decoded by several JPEG2000 Part-1 compliant decoders. It means that the proposed method can meet the syntax compliance requirement. Experimental results also show that the information-embedded codestream can also be correctly decoded after truncating some packets. That is, the proposed method can also meet the scalability requirement.

It is worth mentioning that the operations of information embedding and extraction are performed at packet level. It means that very few decoding operations are involved in the proposed method. That is, the proposed method can meet the simplicity requirement for naïve information hiding. With these advantages, the proposed naïve JPEG2000 information hiding method can be used to embed useful information for practical JPEG2000 applications.

Acknowledgements

This work was supported partially by the National Science Council of Republic of China under grant NSC 97-2221-E-606-017.

References

[1] S. Katzenbeisser and F.A.P. Petitcolas, I*nformation hiding techniques for steganography and digital watermarking*, Artech House, Boston, 2000.

[2] D.S. Taubman and M.W. Marcellin, *JPEG2000 - Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, Boston, 2001.

[3] M. Rabbani and R. Joshi, "An overview of the JPEG 2000 still image compression standard," *Signal Processing: Image Communication*, vol. 17, no. 1, pp. 3-48, 2002.

[4] Information technology - JPEG 2000 image coding system, ISO/IEC International Standard 15444-1, ITU Recommendation T.00, 2000.

[5] H. Noda, J. Spaulding, M.N. Shirazi, and E. Kawaguch, "Application of bit-plane decomposition steganography to JPEG2000 encoded images," *Proceedings of the 2002 IEEE International Conference on Image Processing*, vol. 2, pp. II-909 - II-912, Sept. 2002.

[6] H. Noda, J. Spaulding, M.N. Shirazi, and E. Kawaguchi, "Application of bit-plane decomposition steganography to JPEG2000 encoded images," *IEEE Signal Processing Letters*, vol. 9, no. 12, pp. 410-413, Dec. 2002.

[7] P.-C. Su and C.-C. J. Kuo, "Steganography in JPEG2000 compressed images," *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 824-832, Nov. 2003.

[8] J. Chen, T.-S. Chen, and C.-Y. Cheng, "A new scheme of image data hiding based on EBCOT of JPEG2000 lossy compression," *Proceedings of the 2004 IEEE International Conference on Networking*, Sensing and Control, vol. 2, pp. 990-995, 2004.

[9] H. Noda, T. Furuta, M. Niimi, and E. Kawaguchi, "Application of BPCS steganography to wavelet compressed video," *Proceedings of the 2004 IEEE International Conference on Image Processing*, vol. 4, pp. 2147-2150, Oct. 2004.

[10] W. Liu, "Data hiding in JPEG 2000 code streams," Proceedings of the 2004 IEEE International Conference on Image Processing, Vol. 3, Oct. 2004, pp. 1557 – 1560.

[11] G. Xuan, D. Jiang, H. Ji, Y.Q. Shi, D. Zou, L. Liu, H. Liu, and W. Bai, "Identity verification system using data hiding and fingerprint recognition," *Proceedings of the 2005 IEEE 7th Workshop on Multimedia Signal Processing*, pp. 1-4, Oct. 2005.

[12] Z. Liang, "Wavelet domain steganography for JPEG2000," *Proceedings of the 2006 IEEE International Conference on Communications, Circuits and Systems*, vol. 1, pp. 40-43, Jun. 2006.

[13] R. Grosbois and T. Ebrahimi, "Watermarking in the JPEG 2000 domain," *Proceedings of the 2001 IEEE Fourth Workshop on Multimedia Signal Processing*, pp. 339-344, Oct. 2001.

[14] P.-C. Su, H.-j. M. Wang, and C.-C. J. Kuo, "An integrated approach to image watermarking and JPEG-2000 compression," *Journal of VLSI Signal Processing*, vol. 27, pp. 35-53, 2001.

[15] K. Li and X.-P. Zhang, "An image watermarking method integrating with JPEG-2000 still image compression standard," *Proceedings of the 2003 IEEE Canadian Conference on Electrical and Computer Engineering*, vol. 3, pp. 2051-2054, May 2003.

[16] A. Makhloufi, A.O. Zaid, R. Bouallegue, and A. Bouallegue, "Watermark integration to wavelet image coding scheme," *Proceedings of the Eighth IEEE International Symposium on Multimedia*, , pp. 685-689, Dec. 2006.

[17] M. Kurosaki, K. Munadi, and H. Kiya, "Error correction using data hiding technique for JPEG2000 images," *Proceedings of the 2003 IEEE International Conference on Image Processing*, vol. 3, Sept. pp. III-473-III-46, 2003.

[18] R. Grosbois, P. Gerbelot, and T. Ebrahimi, "Authentication and access control in the JPEG 2000 compressed domain," *Proceedings of the SPIE 46th Annual Meeting, Applications of Digital Image Processing XXIV 4472*, pp. 95–104, 2001.

[19] IrfanView, http://www.irfanview.com/

[20] JJ2000, http://jj2000.epfl.ch/

[21] PhotoImpact 10, http://www.ulead.com.tw/

[22] ACDSee 7.0, http://www.acdsee.com/.