EUSIPCO 2013 1569743443

FORMAT COMPLIANT ROI ENCRYPTION OF JPEG XR BITSTREAMS BASED ON TILING

Jutta Hämmerle-Uhl, Stefan Jenisch, and Andreas Uhl

Multimedia Signal Processing and Security Lab (WaveLab)
Department of Computer Sciences, University of Salzburg
J.-Haringerstr.2, 5020 Salzburg, AUSTRIA
andreas.uhl@sbg.ac.at

ABSTRACT

Tiling is used to facilitate format compliant region of interest encryption of JPEG XR codestreams. The proposed technique can be used with any desired cryptographic cipher and avoids decoding problems by modifying the tile index of the image in order to prevent a decoder to process the encrypted parts. Experimental examples are given and the impact of tiling on compression performance is assessed.

Index Terms— JPEG XR, Region of interest encryption, format compliance

1. INTRODUCTION

Region of interest (RoI) encryption has been applied successfully e.g. to selectively protect parts of the visual data in surveillance videos (i.e. faces in most cases, which results in privacy enhancing technology (PET) for such schemes [1, 2]) or to protect parts of medical imagery to protect patients privacy and anonymity [3]. Existing approaches for region of interest image and video encryption can be classified by the domain in which they perform the encryption, as done in related surveys on selective encryption [4]: Encryption can either be applied to the data before (pre-compression), during (in-compression), or after coding (post-compression) [5]. The first two approaches suffer from eventual impact on compression performance, and the in-compression scheme additionally does usually not allow to use existing compression hardware. Format compliance is easily achieved with those two techniques, however, quite often, the RoI is not known before or during compression. The post-compression approach on the other hand is not limited by any impact on compression and is highly flexible since it is based on a generic bitstream. However, format compliance is an important issue which can not be achieved easily for the encryption schemes employed (see e.g. related techniques for JPEG2000 [5]).

The encryption of data (to be) encoded with JPEG XR [6] is entirely in its infancy. Two in-compression encryption techniques have been applied to provide PET for video surveillance using a RoI encryption technique [7, 2]. The general limitations of the in-compression approach of course also do apply here: There is impact on compression efficiency, existing JPEG XR hardware [8] cannot be used, and the RoI to be protected needs to be known when the data is compressed.

In this work, we propose a technique to apply RoI encryption to JPEG XR codestreams maintaining format compliance. Thus, the image data encrypted by the suggested approach can be displayed by any JPEG XR decoder / viewer, but the plaintext data of the RoI is replaced by ciphertext information not revealing the plaintexts visual content in this area. The encryption process is a post-compression one, i.e. encryption is applied to the JPEG XR codestream. In Section 2, we shortly review the basic principles of JPEG XR. Section 3 explains first how we are able to achieve generic post-compression JPEG XR encryption and additionally extends the idea to result in RoI encryption. In Section 4 we provide experimental results and discuss the techniques introduced. Section 5 concludes the paper.

2. JPEG XR

JPEG XR has been originally derived from the Microsoft HD Photo format and is primarily designed for the efficient compression of still tone images. One of the primary design goals was to provide support for up to 32 bits per colour channel which allows for high dynamic range imaging and true lossless compression. Furthermore the standard aims to achieve a better compression ratio than the JPEG baseline standard in the low bitrate range and is also designed for minimal computational requirements and ease of hardware implementation which comes apparent when looking at the design of the block-based frequency transformation process. The process is DCT based, relies completely on integer operations and allows for a lifting scheme like implementation. The standard also supports tiling, which allows the image to be partitioned into independently processable segments. Tiling will be used in this work to address RoI encryption.

The decoding process involves several steps and can be

This work has been partially supported by the Austrian Research Promotion Agency FFG, Bridge project no. 832082.

divided into image parsing and image decoding. In Figure 1 a graphical presentation of the process is shown.

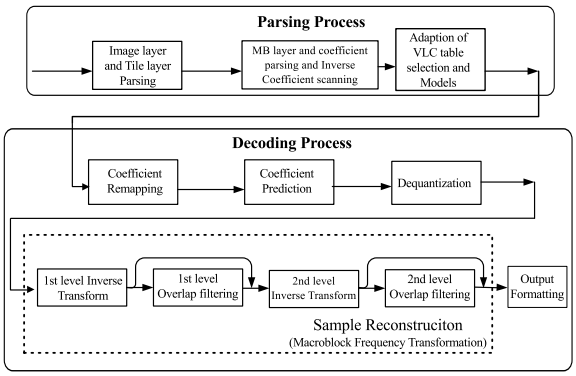

Fig. 1. The JPEG XR parsing and decoding process (Source: ITU-T Standard T.832, page 43)

JPEG XR uses a block-transform for signal decorrelation. The coefficients are grouped into three frequency bands, the DC, Lowpass- (LP) and Highpass (HP) bands, respectively. The three frequency bands result from the frequency transformation of the Macroblocks. For frequency transformation of an image, it is partitioned into a set of Macroblocks having the size of 16×16 pixels. These Macroblocks are then transformed using the schema shown in Figure 2 by decomposing the Macroblock into atomic transform blocks of a size of 4×4 pixel. The process has two stages. First these atomic blocks are transformed into the frequency domain. Second the DC portions of these blocks are grouped together and again frequency transformed. The result of this process is one DC, 15 LP and 240 HP coefficients for each Macroblock. The frequency transform used in JPEG XR is the Photo Core Transform (PCT).

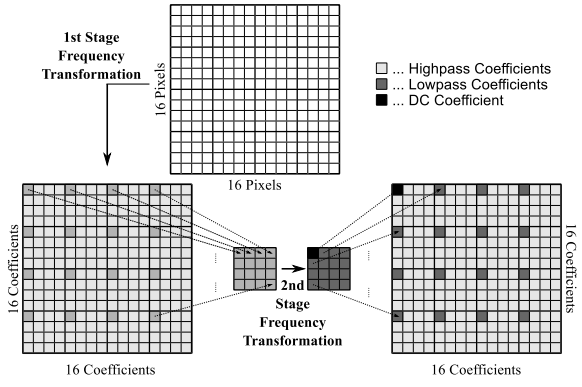

Fig. 2. The Macroblock and its transformation into the frequency domain.

To address the blocking effect a second block transformation beside the PCT was defined which is called "Photo Overlay Transform" (POT). The POT is optionally applied to the image but its transformation grid is shifted vertically and horizontally by two pixels relative to the PCT grid.

The JPEG XR standard allows the organisation of the code-stream in two different modes within each tile. The organisation in "Spatial Mode" and the organisation in "Frequency Mode". The code-stream structure is shown in Figure 3.

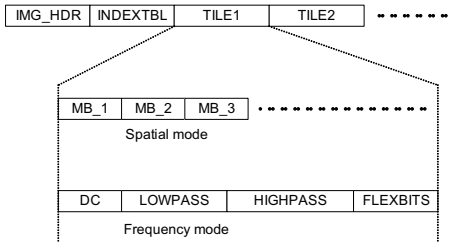

Fig. 3. JPEG XR code-stream structure (Source: ITU-T Standard T.832, page 43)

In both modes the image data is organised according to the tile sequence (from left to right and top to bottom) and preceded by the image header information and the index table (containing offset positions of the tiles in the codestream). The modes differ in the way the data is organised inside the tiles.

In "Spatial Mode" the data is organised according to the Macroblock sequence from left to right and top to bottom.

In "Frequency Mode" the data is ordered according to its frequency band. After the two-dimensional frequency transformation the resulting coefficients are grouped together in three frequency bands (DC, LP, HP) which are than transmitted according to their importance starting with the low frequency portions. This allows for a low quality preview image during transmission similar to progressive JPEG.

3. FORMAT-COMPLIANT JPEG XR ROI ENCRYPTION

While format compliant encryption may be defined by requiring all syntax definitions of the file format to be obeyed after the encryption process, an alternative definition is to require the reference software to be able to decode the encrypted bitstream properly (without decoding errors). For practical reasons we stick to the second definition here, the proposed approach is only format compliant in the latter sense but not format compliant in the syntactical sense.

Encrypting the encoded image data on a bitstream level using a block or stream cipher usually breaks format compliance of the resulting file. This is also true when encrypting the JPEG XR codestream in such a way. Even when leaving the header information untouched and only encrypting the frequency bands (coefficient related data) it is most likely that the decoding will fail because the encrypted data will mis-

guide the variable length decoding engine. This usually results in an error condition of the next decoding stage of the pipeline (e.g. 18 coefficients are assigned to a transform block of the size 4×4 pixels). So the encrypted codestream is not format compliant.

We use a method which allows to encrypt the coefficient data on the frequency band level (DC, LP, HP) with an arbitrary block / stream cipher but still remaining format compliant, termed *Encrypting Entire Frequency Bands* (EEFB).

When using frequency store mode the JPEG XR standard requires that there is an index entry at the beginning of the file which contains the positions locating the frequency bands DC, LP, HP and FLEXBITS in the code stream. These positions or offsets are used to locate the encoded coefficients during decoding.

The strategy is now to encrypt the coefficient data of a frequency band (e.g.: DC, LP or HP) with any given cipher. Subsequently, we generate a dummy data block used to mimic coefficient data and put this block in front of the encrypted coefficient data in the code-stream. Additionally, the tile index has to be modified respectively to point to the dummy data blocks. These modifications prevent the decoding engine from decoding the encrypted data since the dummy data are decoded and the subsequent encrypted parts are ignored. In Figure 4 the principle is visualised.

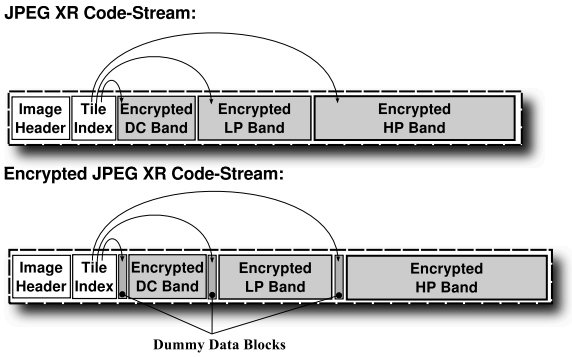

Fig. 4. Encrypting JPEG XR code-streams using dummy data blocks.

The major drawback of the method might be the increased file size due to the dummy data blocks. Fortunately, these dummy data blocks do not require much space if "sparse" frequency bands are used for representing them. In these data blocks all coefficients are set to zero. The JPEG XR standard defines coded-block patterns which makes encoding of "sparse" frequency bands quite efficient. These coded-block patterns are positioned at the start of an encoded Macroblock and indicate (for all colour channels) if a coefficient is non-zero in the Macroblock. So for a dummy data block just a sufficient number of coded-block patterns has to be generated, indicating all coefficients are zero in a corresponding Macroblock.

For the DC frequency band a coded block pattern having 2 bits is needed to indicate a Macroblock where all coefficients (for all colour bands) are zero. Also the LP band requires 2 bits for each Macroblock (indicating that all 15 LP coefficients in all colour bands are zero) while only one bit per Macroblock is needed for the HP frequency band (indicating that all 240 HP coefficients in all colour bands are zero). The FLEXBITS can be encrypted without putting a dummy data block in front since there is no VLC involved. Since this segment of the code stream is only parsed by the program code responsible for the FLEXBITS, possible generation of marker bits poses no problem.

Setting the dummy data block to zero as described offers two advantages: First, file size is minimised this way. Second, this strategy reproduces the best result of a "Replacement Attack". As the name already suggests, this attack replaces encrypted data with NULL- or averaged data (in [9] this attack was described for bit plane encryption). The alternative of producing a dummy data block containing disturbing image artefacts would strengthen visual security but at the cost of an increased file size. Additionally, a skilled attacker would simply run a "Replacement Attack" anyway (thus removing the artefacts causing noise in the image).

This is why encrypting all frequency bands (DC+LP+HP) produces a flat grey image with the proposed settings. In Figure 5 the impact on image quality of this encryption method can be seen when decoding the encrypted images without key.

Fig. 5. Decoding the protected file: Encrypted DC-, LP-, HP- and DC+LP band.

The fundamental idea of EEFB can be immediately used to accomplish RoI encryption using the tiling mechanism of JPEG XR. The strategy starts by overlaying an equidistant grid to an image which can be transformed into JPEG XR compliant tiles (so grid lines have to be aligned to Macroblock boundaries). Subsequently, the tiles corresponding to the RoI to be protected are encrypted with any desired cipher, the remaining ones are left in plaintext. By analogy to EEFB, the tile index of the image is then modified in such a way that the offset entries for the encrypted tiles point to (a) dummy tile(s) which can be decoded properly but do not at all correspond to the visual information of the RoI. In Figure 6 the principle is visualised.

There are two options to generate the dummy tile(s): First, the tile can be merely a constant grey (or any other colour) tile of minimal size which is inserted into the code stream. Of course, this has a slightly negative impact on the code stream size. Contrasting to EEFB, in the RoI encryption approach the

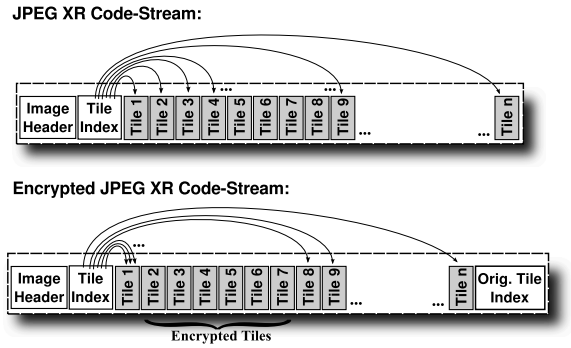

Fig. 6. RoI encryption of JPEG XR code-streams using dummy tiles.

actual insertion of dummy data can be avoided entirely. The second option thus is to modify the tile index in a way that for encrypted tiles it points to unencrypted plaintext tiles showing image content which does not need to be encrypted. In this manner, unencrypted tiles are actually "duplicated" over encrypted ones. Figure 7 visualises this approach where the upper left tile of the Lena image is duplicated over all tiles of the facial area. Note that the encrypted tiles are of course present in the codestream (which is not format compliant in the syntactical sense, but decoding ignores those tiles and replaces them with duplicates of plaintext tiles).

Fig. 7. RoI encryption of Lena facial region using plaintext tile duplication (using 16×16 tiles).

It is of advantage to insert the original tile index into the code-stream (eventually in encrypted manner) as this eases the decryption process since there is no need to rebuild the tile index by parsing and interpreting the code-stream. A good position to store an eventual additional dummy tile and the encrypted original tile index is in front of the first valid tile,

Encrypted Frequency Band:	File Size:	CCR:
Unmodified image:	461105 kbyte	—
encrypted DC band:	461379 kbyte	0.06%
encrypted LP band:	461313 kbyte	0.05%
encrypted HP band:	461239 kbyte	0.03%
encrypted DC+LP band:	461587 kbyte	0.10%
encrypted DC+HP band:	461513 kbyte	0.09%
encrypted LP+HP band:	461447 kbyte	0.07%
encrypted DC+LP+HP band:	461721 kbyte	0.13%

Table 1. Changed file sizes for the Lena image when adding dummy data blocks.

although this requires to update all offsets in the new index table. Alternately, JPEG XR comment fields or the end of the codestream can be used to accommodate the encrypted original tile index. Here, the presence of encrypted tiles and their location can be signalled as well.

4. EXPERIMENTS AND DISCUSSION

File sizes when applying EEFB in various settings to the Lena image can be found in Table 1, the increase is significantly below 1 % when quantisation is turned off. Naturally, quantisation worsens the effect since the process decreases the original file size but the empty frequency bands stay at the same size. Still the overall filesize increase remains manageable also with significant compression.

In terms of filesize increase, the encryption of the DC band is most costly since the empty frequency band is of the biggest size when comparing it to the LP and HP frequency bands. On the other hand, the encryption of the HP band is the most expensive in terms of CPU usage since the HP band has the most coefficients.

The main drawback of the RoI encryption approach is that tiling in general has a negative influence on compression performance (on the other hand tiling makes the bit stream more error resilient, it allows for parallelised processing, and it allows for ROI functionality). While compression impact is also true for tiling in JPEG2000 (see [5] for the impact of tiling on coding performance in the RoI encryption context), the reason for this effect is entirely different in JPEG XR. In JPEG2000 tiling results in the fact that the wavelet transform is applied to single tiles instead of applying it to the entire image. This results in reduced energy concentration of the transform thus reducing overall compression efficiency. In JPEG XR, the transform is applied on a Macroblock basis in any case, thus, as long as tiles are not smaller than Macroblocks (16×16 pixels) the transform is not at all affected. However, the coefficient prediction structure of JPEG XR is changed. DC and LP coefficients are predicted from adjacent Marcoblocks, but only within a single tile. Therefore, when introducing tiles, an increasing number of DC and LP

coefficients are no longer predicted for a decreasing tile size, thereby reducing compression efficiency.

In Figure 8 the impact of tiling on the actual compression performance for different quantisation settings can be seen (CCR denotes changed compression ratio, i.e. file size increase in %). The image used to generate the plot is the Lena image with size 512×512 pixel.

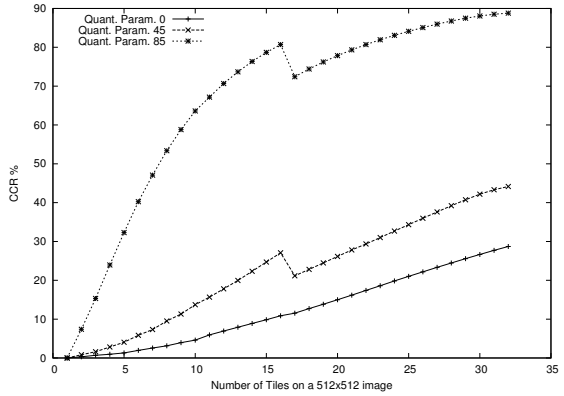

Fig. 8. The impact on compression performance when using tiles. The buckling in the curves is due to a change in the grid layout.

The first observation is that more severe compression suffers more from tiling. While the file size increase is moderate for the lossless case (up to 30%), it grows up to almost 90% for strong compression settings. The second observation is a continuous degradation of compression performance for increasing the number of tiles. These results confirm first investigations with respect to tiling impact on compression performance for JPEG XR [10]. As a consequence, we face a tradeoff between accuracy of the RoI coverage (small tiles are required to cover arbitrarily shaped RoIs) and compression performance.

A final remark refers to the general applicability of the technique. While it is correct that this approach can be applied to any given JPEG XR codestream (thus the RoI position and shape does not need to be known prior to compression), tiling has to be applied in the compression stage to facilitate employment of the proposed technique. Thus, some impact on compression performance cannot be avoided even in case no RoI encryption is performed.

5. CONCLUSION

Format compliant RoI encryption of JPEG XR codestreams can be facilitated using the tiling mechanism of JPEG XR in an elegant manner. The advantages of the approach are that the RoI information does not need to be available at compression time and that the encryption can be applied to the codestream using any (cryptographically strong) encryption technique without breaking format compliance (i.e. without causing a decoder to fail). Thus, security and computational complexity carry over from the encryption scheme used.

The downside of the approach is an intrinsic impact on compression performance due to the employed tiling mechanism. Furthermore, format compliance is not provided in a strict syntactical sense.

6. REFERENCES

[1] Frederic Dufaux and Touradj Ebrahimi, "Scrambling for privacy protection in video surveillance systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1168–1174, 2008.

[2] Hosik Sohn, W. De Neve, and Yong Man Ro, "Privacy Protection in Video Surveillance Systems: Analysis of Subband-Adaptive Scrambling in JPEG XR," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 2, pp. 170–177, Feb. 2011.

[3] Yang Ou, Chul Sur, and Kyung Hyune Rhee, "Region-based selective encryption for medical imaging," in *Proceedings of the International Conference on Frontiers in Algorithmics (FAW'07)*, Lanzhou, China, Aug. 2007, Lecture Notes in Computer Science, pp. 62–73, Springer-Verlag.

[4] Dominik Engel, Thomas Stütz, and Andreas Uhl, "A survey on JPEG2000 encryption," *Multimedia Systems*, vol. 15, no. 4, pp. 243–270, 2009.

[5] Jutta Hämmerle-Uhl, Rudolf Schraml, and Andreas Uhl, "Privacy enhancing technologies in video surveillance applied to JPEG2000 codestreams," in *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP'12)*, Sept. 2012, pp. 95–100.

[6] Frederic Dufaux, Gary J. Sullivan, and Touradj Ebrahimi, "The JPEG XR image coding standard," *IEEE Signal Processing Magazine*, vol. 26, no. 6, pp. 195–199, Nov. 2009.

[7] H. Sohn, W. DeNeeve, and Y.M. Ro, "Region-of-interest scrambling for scalable surveillance video using JPEG XR," in *ACM Multimedia 2009*, Beijing, China, Oct. 2009, pp. 861–864.

[8] C.-H. Pan, C.-Y. Chien, W.-M. Chao, S.-C. Huang, and L.-G. Chen, "Architecture design of full HD JPEG XR encoder for digital photography applications," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 3, pp. 963 –971, Aug. 2008.

[9] A. Uhl and A. Pommer, *Image and Video Encryption. From Digital Rights Management to Secured Personal Communication*, vol. 15 of *Advances in Information Security*, Springer-Verlag, 2005.

[10] J. De Cock, C. Hollemeersch, J. Wielandt, and P. Lambert, "Ultra high definition video decoding with Motion JPEG XR using the GPU," in *Proceedings of the IEEE International Conference on Image Processing (ICIP'11)*, Brussels, Belgium, Sept. 2011, pp. 377–380.