https://doi.org/10.2352/ISSN.2470-1173.2018.07.MWSF-370

Image Scramble Algorithm with Robustness under Transcoding

Chanyul Kim, In kwon Choi, Minwoo Park, Kwangpyo Choi, Jeonghoon Park; Samsung Research; Seoul, South Korea

Abstract

In this work, image scramble algorithm is proposed that is robust under transcoding to recover a successfully descrambled image from a scrambled image. The transcoding schemes as such as resizing, color space conversion, color format conversion, etc. are considered as they are commonly applied in multimedia sharing Instant Messaging services (IMs) applications. To combat such transcoding, following methods are applied: MCU block shuffle scramble, restricted DCT coefficient sign randomization, restricted DCT coefficient swap within MCU block, adaptive DCT coefficient scaling and clipping depending on the image and post-update. The resulting scrambled image yields good visual scramble effects and small increase in bitstream size compared to the original image. The scramble key, that scramble algorithm depends upon, is encrypted with the authorized user information and embedded in the scrambled image, chosen prior to transmission, to successfully descramble the scrambled image without additional servers for key management. These results allow scrambled image to be shared via IMs and descramble successfully without key management servers, thereby possibly creating new multimedia service for smartphones.

1. Introduction

With the rapid widespread of smart phone devices with camera, cheaper network data fee and instant messaging services (IMs), more people are sharing multimedia data (especially JPEG images) with each other via IMs. Also with the recent burst of public interest in security due to exposures on government surveillance and invasion of privacy, providing secure multimedia and method of sharing multimedia information between trusted parties is an important problem that needs to be addressed. In order to be shared via IMs, the secure multimedia data must conform to the multimedia standard format and traditionally encrypted multimedia data often violates the standard format and is therefore unable to be shared via IMs. As JPEG images are one of the most shared multimedia format via IMs, we will focus on JPEG image scramble algorithms.

Three important criteria of image scramble algorithm are as follows: provide sufficient visual scramble effect, suppress bit stream size expansion and ensure sufficient image quality of the correctly descrambled image that approximates the original image. There has been notable approaches and recent advances on the topic of scramble algorithms combining DCT coefficients sign randomization, DCT coefficient permutation and segmented DC grouping [1] that satisfies the above three criteria well enough. However, when scrambled image is shared via IMs, transcoding is applied during transmission and the scrambled image is modified along with introduction of error/noise from transcoding. In addition, IMs transcoding usually deletes any unnecessary meta-data in the JPEG file prior to transmission to reduce the size of the transmitted data handled by the IMs servers. Any extra bit-stream data after the End-Of-Image JPEG marker is thrown away, along with any irrelevant meta-data as such as application specific Exchangeable-Image-File-Format (EXIF) data, thumbnail images, comments, etc. Some IMs transcoding completely reformats the image by full decoding to RGB data and then re-encoding under strict conditions as such as color format (e.g. YUV4:2:0), standard quantization table, standard Huffman table and baseline sequential format. Therefore hiding information that assist in the scramble/descramble process as a meta-data or as a hidden bit-stream is not a good practice as there is no guarantee of its survival after IMs transcoding.

Also in IMs sharing environment, the image is sent once by the sender and recipients receive copy of the same image. In addition, key management servers would be too costly to set up and manage, therefore sharing of the scramble key between sender and recipients are preferred to be done in P2P environment or embedded in the scrambled image. So this work, image scramble algorithm that is robust under transcoding and is able to support authorized users without key management servers, is presented.

2. Transcoding Bottlenecks

In this section, the transcoding bottlenecks of IMs image sharing that needs to be overcome by the scramble/descramble algorithm are identified.

First bottleneck is possible YUV saturation arising from inverse discrete cosine transformation (DCT) of scrambled unnatural DCT frequency coefficients when JPEG decoding is executed by the transcoder. This saturation leads to unrecoverable loss of information and must be minimized for successful descrambling.

Second bottleneck is possible RGB saturation arising from YUV to RGB conversion of scrambled YUV values executed by the transcoder. Most of the IMs transcoders fully decode the JPEG image to RGB pixel values before re-encoding again, therefore YUV to RGB and then RGB to YUV conversions take place in most IMs transcoders. This RGB saturation also leads to unrecoverable loss of information and must be minimized.

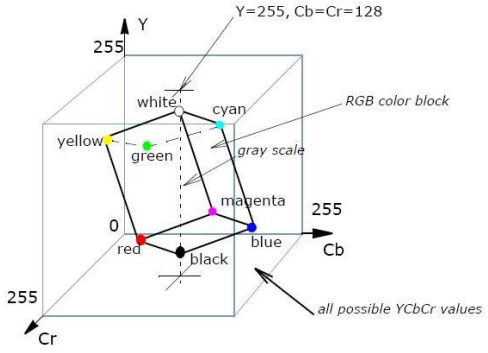


Figure.1. *RGB color cube inside of corresponding larger YCbCr (YUV) cube. YUV values outside of the RGB cube are "invalid."*

Figure.1 illustrates the 8-bit RGB cube inside the corresponding 8-bit YCbCr cube. The Y = U = V = 128 coordinate is the color "grey" and equivalent to R = G = B = 128 coordinate at

the center of the cube. Notice that color "white," which has R = G = B = 255 coordinate, has Y value slightly less than 255 and U = V = 128. Figure.1 clearly illustrates that 8-bit YUV space is bigger than 8-bit RGB space in terms of color expression. The "invalid" YUV coordinates outside of the RGB color cube has no corresponding "valid" RGB coordinates. During YUV to RGB conversion, any "invalid" YUV is mapped to "invalid" RGB values and clipping to be within the range of 0 to 255 for 8-bit precision. The clipping procedure is transcoder implementation dependent and therefore it can yield different outcomes for different IMs. Also, if scrambled YUV value is "invalid" YUV value, then after the transcoding procedure of YUV to RGB conversion, followed by clipping and RGB to YUV conversion, the scrambled YUV values will have changed to a different value and will not be able to descramble back to the original YUV.

Third bottleneck is color format conversion, for example to YUV4:2:0 formats that are often exclusively supported by the IMs. Pre-processing of the original image to the default color format supported for the IMs transmission may provide solution.

Final bottleneck is complete re-encoding of the image with new quantization table and Huffman table decided by the IMs. Some measures to anticipate this by the scramble algorithm is vital, as any dependencies on quantization table and Huffman table of the scramble algorithm may severely affect recoverability to the original image.

3. Scramble Algorithm

3.1 Existing Problems

Previous approaches in image scrambling algorithms include sign randomization [2], permutations of DCT coefficients [3] [4] and combination of above with DC permutation via segmentation [1]. While [2] has an advantage of negligible bit-stream expansion, it suffers from attacks described in [4], where original image outline can be extracted. While [3] and [4] provide certain robustness against attacks, they suffer from significant bit-stream expansion. As [1] illustrates possible attacks using Nonzero-Coefficient-Count (NCC), Energy-of-AC-Coefficients (EAC) and Position-of-Last-Nonzero-Coefficients (PLZ) for intra-block based image scramble algorithms, some global shuffling of blocks is required in a successful scramble algorithm to prevent these types of attacks. While [1] combines previous approaches to achieve some degree of improvement, it does not take transcoding bottlenecks as such as YUV and RGB saturation into consideration and is not robust under most IMs transcoding.

3.2 Building blocks of proposed algorithm

For each minimum coded block (MCU), an inter-MCU scramble operation and intra-MCU scramble operations are applied in sequence. This enables fast scramble operations as scrambling is done on-the-fly as the image is being decoded or encoded.

An inter-MCU scramble operation consists of MCU queue shuffling depending on some scramble key based on pseudorandom sequences. Each MCU is assigned randomly, depending on the scramble key, a group number out of N groups in total. Then MCU are queued in each group according to their group number and in raster scan order of the original image. So while scanning MCUs in raster scan order in the original image, the first MCU with group number 1 is the first in the queue for Group 1 and the next MCU with group number 1 assigned is the second in the queue for the Group 1. When all of the MCUs with group number 1 have been queued, we move on to Group 2 with MCUs with group number 2 being queued. This process is repeated until Group N completes the queuing process. Then first group queue (Group 1) forms the first section of the new raster scan order of the shuffled image and the second group (Group 2) forms the second section following the first section and etc. until the last N group queue forms the last section of the new raster scan order of the shuffled image. Figure.2 illustrates an example of MCU queue shuffle with 16 MCUs and 4 groups. This inter-MCU scramble algorithm of MCU queue shuffle provides good scramble effect with controlled bitstream size expansion as some correlation arising from maintained queue orders is not destroyed and provides sufficient compressible redundancies. This MCU queue shuffle is completely reversible to obtain the original image as long as the same scramble key assigns the group numbers and its detail is omitted here.

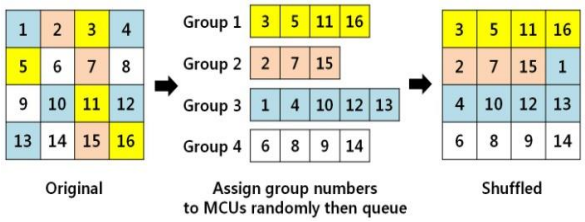


Figure.2. *Inter-MCU scramble algorithm: MCU Queue Shuffle.*

An Intra-MCU scramble operations consist of six scramble algorithms that depend on some scramble key based on pseudorandom sequences. First three are specific types of restricted DCT frequency coefficient sign randomization, which are: Figure.3 (a), 8 by 8 block-wise sign flip; Figure.3 (b), sign flip of even columns within 8 by 8 blocks; Figure.3 (c), sign flip of even rows within 8 by 8 blocks. Fourth is a specific type of restricted DCT coefficient permutation within 8 by 8 blocks, which is swaps of coefficients in (x, y) position with coefficients in (y, x) position, where (x, y) denotes position of coefficient in x^{th} column and y^{th} row. Figure.4 illustrates this restricted DCT coefficient permutation graphically, where, for an example, coefficient labeled "2" is moved from position (2, 1) to (1, 2) and coefficient labeled "9" is moved from position (1, 2) to (2, 1) after the permutation, thus achieving final effect of swap of "2" and "9" labeled coefficients.

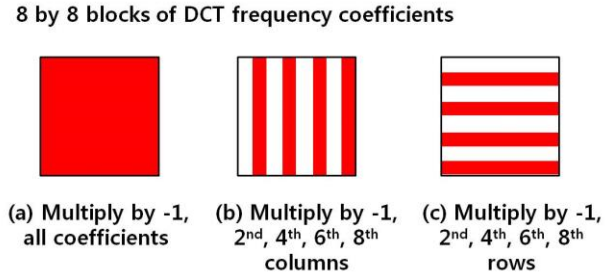


Figure.3*. Restricted DCT frequency coefficient sign randomization.*

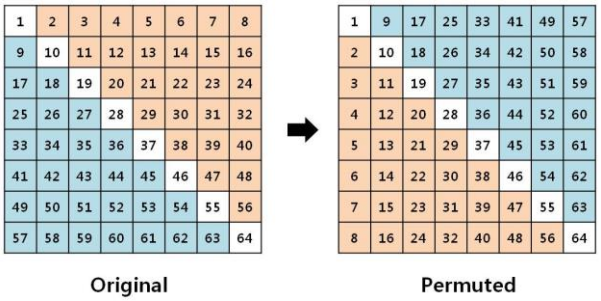


Figure.4. *Restricted DCT frequency coefficient permutation.*

Fifth and sixth intra-MCU scramble algorithms are permutations of 8 by 8 blocks within each of the Y, U and V channels within each MCU. Figure.5 illustrates the two permutations for the case of Y channel in a MCU with YUV4:2:0 formats with four 8 by 8 DCT coefficient blocks. Figure.5 (a) is the fifth scramble algorithm where block 2 is swapped with block 3. Figure.5 (b) is the sixth scramble algorithm where block 1 is swapped with block 2 and block 3 is swapped with block 4.

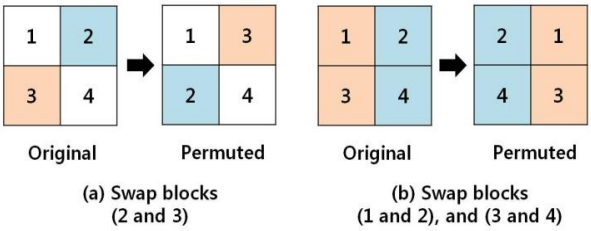


Figure.5. *Restricted 8 by 8 DCT block permutation within MCU.*

3.3 Block-wise sign flip

The first intra-MCU scramble operation, Figure.3 (a), is applied on 8 by 8 block basis for each of YUV channels (4 Y blocks, 1 U block and 1 V block in a MCU for YUV4:2:0 format) depending on the scramble key. This is termed "Block-wise sign flip" and it is applied on each 8 by 8 blocks independently. The scramble key generates pseudorandom sequence of 1s and 0s, and this map determines on which 8 by 8 blocks to apply block-wise sign flip. The end result of this scramble operation is some 8 by 8 blocks with all of the signs of the DCT frequency coefficient flipped and some blocks with no change, that is completely determined by the scramble key. The block-wise sign flip algorithm is chosen because this operation on DCT frequency domain simulates certain operation on YUV values in YUV domain. Applying block-wise sign flip on DCT frequency domain is equivalent to transforming the DCT frequency coefficients to YUV values (inverse DCT), then performing ″inversion″ on YUV values (operation: $x \rightarrow 256 - x$), then transforming back to DCT domain (DCT). Figure.6 shows the formula for DCT implementation suggested in JPEG format. This is needed for the proof that block-wise sign flip simulates a transformation in YUV domain, illustrated in Figure.7. Figure.7 illustrates that starting from the original YUV, applying YUV "inversion" and then applying DCT yields the same result as applying DCT first then applying block-wise sign flip in the DCT frequency domain.

$$G_{u,v} = \frac{1}{4}\alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7}(g_{x,y} - 128)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

where

- u is the horizontal spatial frequency, for the integers $0 \leq u < 8$.
- v is the vertical spatial frequency, for the integers $0 \leq v < 8$.
- $\alpha(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } u = 0 \\ 1, & \text{otherwise.} \end{cases}$
- $g_{x,y}$ is the pixel value [0, 255] at coordinates (x, y).
- $G_{u,v}$ is the DCT coefficient at coordinates (u, v).

Figure.6. *Discrete Cosine Transform(DCT) used in JPEG.*

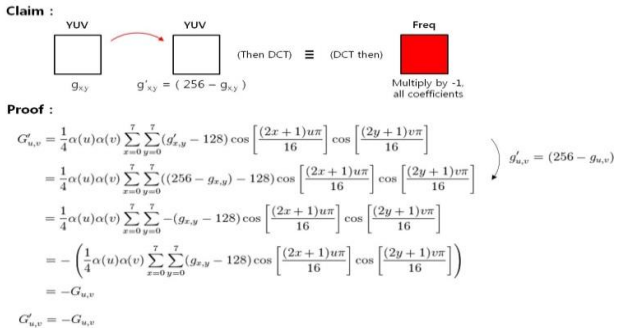


Figure.7. *Proof of equivalence of block-wise sign flip with YUV "inversion".*

This ensures that the block-wise sign flip prevents YUV saturation during transcoding as the scrambled YUV values will be in the range of 0 to 255 and be "valid" YUV values (Except the case of 0 YUV value transforming to 256, which is dealt with by clipping to 255 as a special case in the scramble algorithm).

3.4 MCU-wise symmetric transform

The specific combinations of intra-MCU scramble operations form scramble algorithms in the DCT frequency domain that simulates symmetric transform of corresponding 16 by 16 YUV block in the YUV domain. The second, third and fourth scramble operations work on 8 by 8 DCT blocks and fifth and sixth scramble operations work on MCUs.

Specifically, second, third and fourth scramble algorithms each simulate symmetric transform of 8 by 8 YUV blocks. Figure. 8 illustrates that second scramble algorithm is equivalent to vertical reflection about the center of the 8 by 8 YUV block and Figure.9 illustrates that fourth scramble algorithm is equivalent to diagonal reflection about the center of the 8 by 8 YUV block. (From Figure.4, it can be easily seen that fourth scramble algorithm is the same as diagonal reflection about the center of the 8 by 8 DCT block.) In YUV domain, vertical reflection and diagonal reflection about the center of the 8 by 8 block are the generators of the symmetric transformations of a square (1 identity, 3 rotations and 4 reflections, i.e. dihedral group D4). Therefore second and fourth scramble algorithms can be used as generators to simulate all 8 symmetric transformations of the YUV block. Third scramble algorithm is equivalent to horizontal reflection about the center of the 8 by 8 YUV block and the proof is omitted here, as it can be shown that the third algorithm is equivalent to the composition of fourth algorithm followed by second algorithm and then followed by fourth algorithm. Third algorithm is useful in that instead of applying fourth algorithm twice and second algorithm once, third algorithm (with same complexity as second algorithm) can be applied once for same effect for efficiency

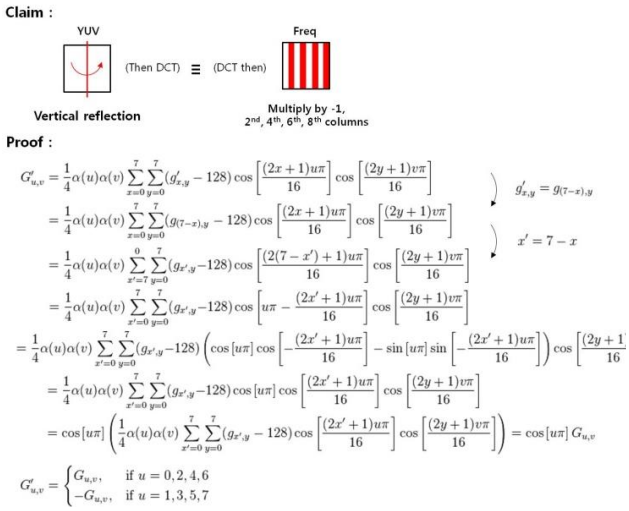


Figure.8. *Proof of equivalence of second scramble algorithm with YUV vertical reflection.*

Therefore the composition of combination of second, third and fourth algorithm achieves equivalent effect of applying symmetric transformations on 8 by 8 YUV blocks. However, for YUV4:2:0 format for an example, if different symmetric transformation yielding scramble operations are applied on four 8 by 8 Y DCT blocks in the same MCU, the single U DCT block and the single V DCT block cannot be accommodated to follow the same symmetric transformation due to different sampling factor.

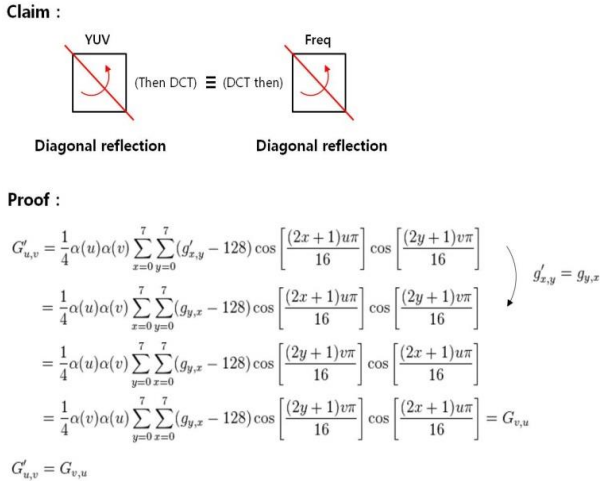


Figure.9. *Proof of equivalence of fourth scramble algorithm with YUV diagonal reflection.*

This often leads to YUV saturation and RGB saturation during IMs transcoding. Therefore the symmetric transformation yielding scramble operations must be applied on MCU as the smallest unit. The fifth and sixth scramble algorithms in conjunction with the second, third and fourth algorithms make it possible to achieve this. Similar to the second and fourth algorithms, the fifth and sixth algorithms form a generator pair for the symmetric transformation of four Y blocks in DCT domain in a MCU. By applying the same combination of second, third and fourth algorithm to all 8 by 8 DCT blocks in a MCU and then applying corresponding combination of fifth and sixth algorithm to permute the 8 by 8 blocks yield a symmetric transformation of the MCU in the YUV domain (16 by 16 YUV block). For example, applying fourth algorithm to each 8 by 8 blocks in the MCU and then applying fifth algorithm on the MCU yields symmetric transformation of the 16 by 16 YUV block of diagonal reflection about the center of the block. These specific compositions of second, third, fourth, fifth and sixth algorithms simulate symmetric transform of the MCU in YUV domain and so is termed "MCU-wise symmetric transformation," (although in DCT domain, the scramble algorithms are not symmetric transformations). Again, these restrictions are to prevent YUV saturation during the IMs transcoding. As with block-wise sign flip, the scramble key generates random map, which determines which MCU-wise symmetric transformation (out of possible 8) to apply to each MCUs in the image.

3.5 Adaptive scaling and clipping

The block-wise sign flip and the MCU-wise symmetric transformation can be combined to improve visual scramble effect. However, the effect of the composition worsens YUV saturation and RGB saturation. Therefore an additional measure is needed to contain the YUV values within some bounds, in the form of DCT frequency coefficient histogram modification. The DCT coefficient distribution histogram modification algorithms applied are: adaptive DCT coefficient clipping and adaptive DCT coefficient scaling, which are applied differently for each of Y, U and V color channels and dependent on statistics of DCT coefficients of the whole image. These algorithms modify the histogram of DCT coefficients and prevent YUV saturation and RGB saturation occurring during transcoding. DCT coefficient scaling has an equivalent effect of YUV scaling centered at the 128 value (instead of 0) as illustrated by Figure.10. Therefore, DCT coefficient scaling with scale factor less than 1 and DCT coefficient clipping with a certain upper bound and a lower bound sufficiently shrinks and modifies the DCT histogram (and YUV histogram, in turn) to minimize YUV and RGB saturation. Too much scaling and clipping severely lowers image quality of descrambled image and insufficient scaling and clipping results in YUV and RGB saturation and defects in descrambled image.

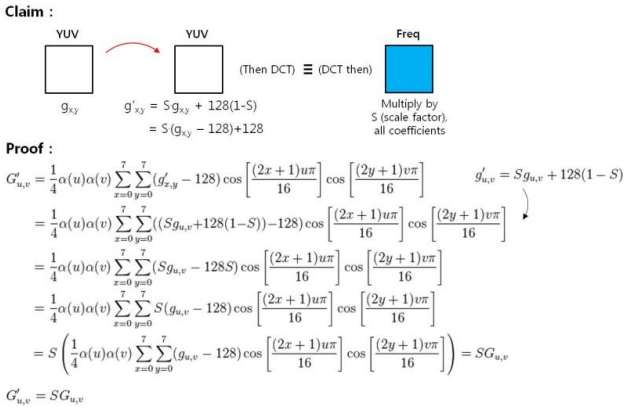


Figure.10. *Proof of equivalence of DCT scaling with YUV scaling.*

As the optimal scale factors and clipping bounds are sensitive to the content and the pixel distribution of the original image, adaptive scaling and clipping are applied to the image, based on

the analysis of the DC values of the Y, U and V color channels across the whole image. Figure.11 illustrates an example of adaptive scale and clipping for Y channel. Figure.11 (a) shows four scale modes for Y channel with different scale factor and clipping bounds. The mode decision is illustrated in Figure.11 (b), where the decision depends on the DC values of Y, U and V channel across the whole of the image and also on scale modes of U and V channels. The data shown in Figure.11 has been trained from test images that yield good image quality of the descrambled images with no defects from YUV and RGB saturation after IMs transcoding. Described operations of DCT coefficient histogram modification are termed "Adaptive scaling" and "Adaptive clipping."

Scale_mode (SM)	Y	
	Scale factor	coef. range
0	1.0	-1024~1024
1	1.1	-988~988
2	1.5	-928~928
3	2.0	-928~928

(a) Scale mode for Y

```
if((DC_CB < 20 || DC_CB > 235) && (DC_Y < 40 || DC_Y > 215)) {
    SM_Y = 3
}
else if(DC_Y > 235)
{
    SM_Y = 1
}
else if(SM_CB != 0 && SM_CR != 0)
{
    SM_Y = 2
}
else
{
    SM_Y = 0
}
```

(b) Scale mode decision for Y

Figure.11. *Adaptive scale and clipping for Y channel.*

3.6 Embedding

The information as such as the scale mode for adaptive scaling and clipping, scramble key and authorized user information must be transmitted with the scrambled image to the intended receiver for the descramble process. To eliminate the need for the key management servers and database management, above information is embedded in the scrambled image. Embedding is done by adding extra rows of MCUs at the top of the image and encoding binary data into each pixel in the added rows of MCUs. In the proposed algorithm, 1 bit of information is encoded in 1 pixel: 0 bit corresponds to pixel value 0 and 1 bit corresponds to pixel value 255. Single scramble key is used to generate needed pseudorandom sequences and random maps for all of the scramble algorithms. This scramble key is then encrypted with authorized user information (i.e. scramble key is encrypted with AES (Advanced Encryption Standard) key of an authorized user) then embedded in the scrambled image. For multiple authorized users, multiple copies of the same scramble key are encrypted with each authorized user's information and secret key, and then embedded into the scrambled image. Identification markers are also embedded so that authorized user can identify which correct encrypted scramble key to decrypt with the secret key.

4. Experimental Results

Kodak 768x512 test images were used in the following experiments. The original test images were subjected to IM transcoding emulation to acquire anchor images. Then original test images were scrambled with proposed scramble algorithm and subjected to IM transcoding emulation and then descrambled to acquire target images. IM transcoding emulation was performed under quantization parameter (QP) of 90, 80, 70 and 60. Figure.12 illustrates two sample sets of original, scrambled and descrambled (after IM emulation) images of Kodak test images. The descrambled images are very good approximations of the original images as can be visually verified in Figure.12. The Bjontegaard-Delta Bit-rates (BD-BR) of the anchor images and target images were calculated and result is shown in Table. 1. The QP range was from 60 to 90 and of 10 samples of Kodak 768x512 test images, average BD-BR was 3.5%. Figure.13 illustrates bitstream size (Kbit) vs. PSNR (dB) graph of Image 3, 7 and 10 from Table. 1, related to the BD-BR calculated. These show that very little image quality loss and defects arise from proposed scramble algorithm.

Bitstream size expansion of the scrambled image and PSNR of the original image and the descrambled image that was scrambled with proposed scramble algorithm are shown in Table 2, showing average bitstream size expansion of 1.3% and average PSNR of 51.b dB. Complexity of the proposed algorithm was measured as follows. The test platform was on Android smartphone using ten 5312x2988 JPEG images shown in Figure.15. Anchor test was normal JPEG decoding, followed by resizing, and then normal encoding. Proposed test was normal decoding, followed by resizing, and then encoding with scramble algorithm to yield scrambled image. Resizing was done to 784x441 to meet potential IMs transmission constraints. Table.3, illustrates the result of complexity comparison between anchor and the proposed scramble algorithm and acceptable average of 8% increase has been measured.

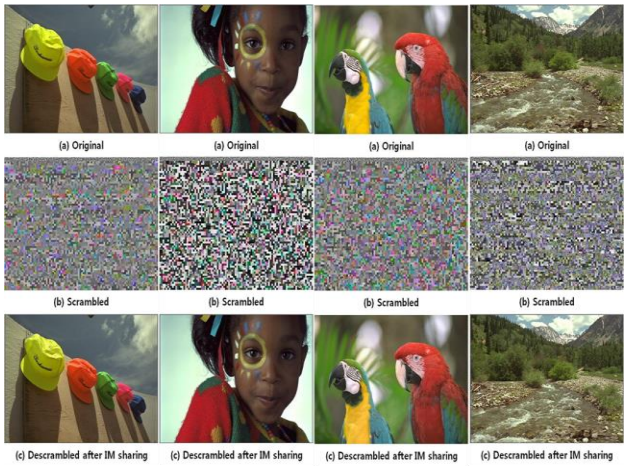


Figure.12. *Comparison of original, scrambled and descrambled images of Kodak 768x512 test set.*

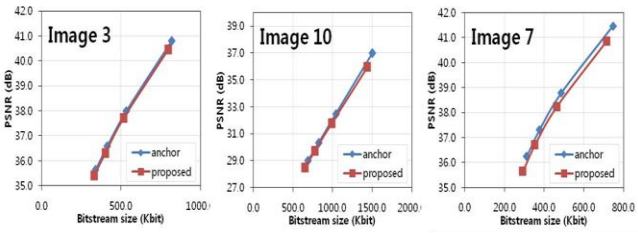


Figure.13. *Bitstream size (Kbit) vs. PSNR (dB) graph of anchor and proposed scramble algorithm from Kodak test images.*

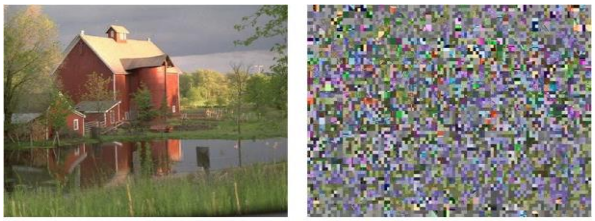
Table. 1. BD-BR (%) comparison of proposed scramble algorithm with anchor of Kodak 768x512 test images.

Image	1	2	3	4	5	6	7	8	9	10	Avg.
Anchor vs. Proposed BD-BR (%)	3.5	2.3	2.2	4.1	3.5	5.5	4.6	3.0	4.2	2.1	3.5

Table. 2. Bitstream size expansion and PSNR results

Image (768x512)	Original	Proposed	Ratio	PSNR		
				Y	U	V
kodak 1	402,572 byte	432,485 byte	7.4%	52.1 dB	52.0 dB	52.0 dB
kodak 2	320,810 byte	341,048 byte	6.3%	51.3 dB	51.7 dB	51.7 dB
kodak 3	276,756 byte	260,322 byte	-5.9%	51.7 dB	41.5 dB	52.1 dB
kodak 4	319,532 byte	346,527 byte	8.4%	52.1 dB	52.1 dB	51.7 dB
kodak 5	416,640 byte	378,630 byte	-9.1%	51.6 dB	52.0 dB	51.7 dB
Average	**347,262 byte**	**351,802 byte**	**1.3%**	**51.8 dB**	**49.9 dB**	**51.9 dB**

The proposed scramble algorithm yielded small bitstream size expansion of average 1.3%, reasonable average BD-BR of 3.5% and small increase of complexity in computation time of 8% on average. Also robustness under transcoding was verified with large test image database and also the embedding scheme with multiple authorized users was verified to be robust under transcoding. There was no failure of authorized descrambling from 4,000 test images that was tested.



(a) Authorised user : Correct scramble key

(a) Unauthorised user : Wrong scramble key

Figure.14. *Bitstream size (Kbit) vs. PSNR (dB) graph of anchor and proposed scramble algorithm from Kodak test images.*

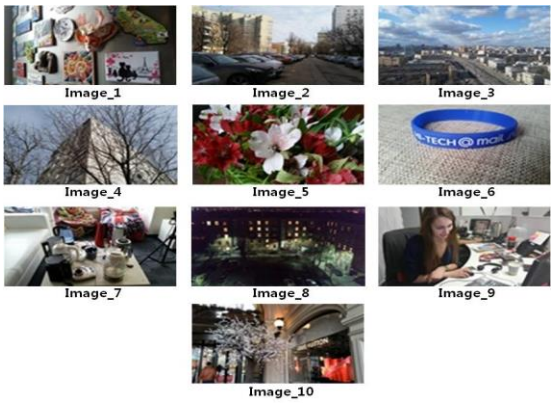


Figure.15. *Test images for complexity measurement*

Table. 3. Complexity comparison measured in computation time.

Image	1	2	3	4	5	6	7	8	9	10	Avg.
Anchor (ms)	272	323	303	428	258	282	270	286	239	335	300
Proposed (ms)	285	354	333	497	281	303	282	303	268	340	325

4. Conclusion

An image scramble algorithm with small bitstream size expansion, good visual scramble effect and good descrambled image quality has been proposed. The increase in complexity due to the proposed scramble algorithm is negligible compared to the complexity of standard JPEG encoders and decoders. Proposed scramble algorithm is compatible with already existing commercial IMs as such as KakaoTalk, Line and WhatsApp. Also the proposed scramble algorithm produces JPEG format compatible scrambled images. Therefore this proposed algorithm is readily applicable into existing frameworks and services.

References

[1] K. minemura. Et al. "JPEG image scrambling without expansion in bitstream size," Image Processing (ICIP), IEEE International Conference, 261–264, Orando, USA, 2012

[2] S.Lian, Z.Liu, Z.Ren, and H.Wang, "Commutative encryption and watermarking in video compression". IEEE Trans. Circuits Syst. Video Technology, **17**, 774-778, 2007

[3] M.Takayama, K.Tanaka, A.Yoneyama, and Y.Nakajima, "A video scrambling scheme applicable to local region without data expansion," In Proc. IEEE ICME, 1349-1352, Toronto, Canada, 2006

[4] W.Li, and Y.Yuan, "A leak and its remedy in JPEG image encryption". Int. J. Comput. *Math*, 1367-1378, 2007

Author Biography

Chanyul Kim received the B.S. degrees from Yonsei University, Seoul, South Korea in 1998 and the PhD degree in computer science from the Dublin City University, Ireland in 2010. He is a principal researcher with Samsung Research in Samsung Electronics, where he has focused on the development of image/video analysis and compression for various Medias.

In Kwon Choi received his BA in Mathematics from the University of Cambridge (2007) and his MSc in theoretical physics from Imperial College London (2009). Since 2011, he was worked at Samsung Electronics Co., Ltd. in Digital Media and Communication R&D Center and AI Center in Samsung Research division in South Korea. His work has focused on media security, video codec and AI for vision recently.

Minwoo Park Min Woo Park received his BS, MS and PhD degrees in computer engineering from Kyung Hee University, in 2003, 2005, and 2009, respectively. He is a senior engineer at Samsung Electronics. His current research interests are moving picture coding, video signal processing, and multimedia systems.

Kwangpyo Choi, Kwang Pyo Choi received the M.S. and Ph.D. degrees in the School of Information Communications Engineering from SungKyunKwan University (SKKU), Korea, in 2000 and 2005, respectively. He is a principal engineer at Samsung Electronics, Research Center. He has been contributing to the developing video codecs and standardizations. His recent interests are image/video technologies using artificial intelligent.

Jeonghoon Park as Vice President of Samsung Research in Samsung Electronics is currently researching the next generation media technology for enabling the future innovative products and services. He has developed the next generation video compression technology by contributing to the international video coding standards such as MPEG-4, H.264 and HEVC of ITU-T and MPEG and currently main focus is to initiate the innovative media products and services by converged cutting-edge technologies such as intelligent media processing, computer vision and data communication