Privacy Protection in JPEG XS: A Lightweight Spatio-Color Scrambling Approach

Takayuki Nakachi[1], Yasuhisa Kato[2], Mitsuru Maruyama[3]
University of the Ryukyus, Nishihara-cho, Okinawa, Japan[1]
Miharu Communications Inc., Kamakura, Kanagawa, Japan[2]
Kanagawa Institute of Technology, Atsugi, Kanagawa, Japan[3]

***Abstract*—This paper presents a lightweight JPEG XS coding scheme incorporating spatio-color scrambling for privacy protection. The proposed approach follows an Encryption-then-Compression (EtC) framework, maintaining compatibility with the JPEG XS standard. Prior to encoding, input images undergo scrambling operations, including line permutation, line reversal, and color permutation. Security analysis indicates that the scrambling technique provides a large key space, making brute-force attacks computationally challenging. Experimental results demonstrate that the proposed method achieves a rate-distortion (RD) performance nearly equivalent to conventional JPEG XS compression while enhancing visual security. Additionally, a rectangular block-based scrambling technique is explored, which offers a trade-off among low latency, reduced memory usage, and visual concealment performance. While real-time processing is possible with or without block-based scrambling, the block-based approach is particularly beneficial for applications that demand lower latency and reduced memory usage. The effectiveness of the proposed method is validated through simulations on 8K ultra-high-definition (UHD) images.**

***Keywords*—JPEG XS; UHD video; Encryption-then-Compression; privacy protection; perceptual scrambling**

I. Introduction

As research into Beyond 5G (B5G) progresses, network and computational infrastructures must evolve to support ultra-low latency, high-speed processing, and intelligent data management. Our research project proposes an architectural framework leveraging in-network computing to facilitate autonomous functional collaboration by seamlessly integrating network and computational resources [1]-[3]. The proposed approach facilitates real-time coordination between networking and computation, optimizing task distribution and adaptive processing while maintaining high throughput and low latency. This is particularly crucial for emerging applications such as real-time ultra-high-definition (UHD) video streaming, where rapid and efficient data processing is essential. In addition, a wide range of real-time applications can benefit from in-network computing, including generative AI, robotics integrated with IoT and sensor technologies, the metaverse, connected vehicles, and digital twins. These domains require ultra-low-latency and high-efficiency processing to support dynamic and data-intensive operations.

One of the key research themes in this project is the utilization of JPEG XS [4]-[9] for high-speed and low-latency video encoding at the edge/cloud while maintaining the quality of the uncompressed video. JPEG XS is an ISO/IEC international standard established in 2019. Similar to JPEG2000 [10], JPEG XS is a coding method based on the wavelet transform. It is known for its low complexity, near-lossless compression, and real-time encoding/decoding capabilities, and is well-suited for applications requiring high-quality, ultra-low-latency video transmission.

A. Existing Challenges and Research Gaps

However, processing data at the edge/cloud and across network infrastructures introduces significant privacy concerns, particularly regarding potential data leakage due to accidental exposure or security breaches [11]-[13]. To mitigate these risks, the Encryption-then-Compression (EtC) framework has been widely explored for privacy-preserving image and video transmission [14]-[29]. Existing EtC techniques have been successfully applied to standardized image coding schemes such as JPEG and JPEG2000. However, despite the recent adoption of JPEG XS as an international standard for ultra-low-latency video encoding, there is currently no dedicated EtC framework optimized for JPEG XS, leaving a critical gap in privacy-preserving video transmission.

Several block-based perceptual encryption schemes have been developed for JPEG and its variations [24]-[29]. However, these conventional techniques are inherently designed for square block-based image coding and are not directly compatible with wavelet-based compression schemes such as JPEG XS. In [22], a block-based JPEG2000 EtC technique incorporating sign-scrambling was introduced. Nevertheless, this method requires a preprocessing step involving discrete wavelet transform (DWT) followed by an inverse discrete wavelet transform (IDWT), introducing additional computational overhead and latency. This preprocessing makes existing approaches unsuitable for real-time applications requiring ultra-low-latency transmission, such as UHD streaming over B5G networks.

B. Contributions of this Study

To bridge this gap, this paper proposes a lightweight scrambled JPEG XS coding scheme designed specifically for privacy-preserving UHD video transmission [1]. The key contributions of our work are as follows:

- A scrambled JPEG XS coding scheme that directly integrates lightweight image scrambling techniques, including line permutation, line reversal, and color permutation, into the JPEG XS encoding process.

[1]Part of this work has been presented at IEEE ISPACS 2022 [30] and IEEE ICICT 2024 [31].

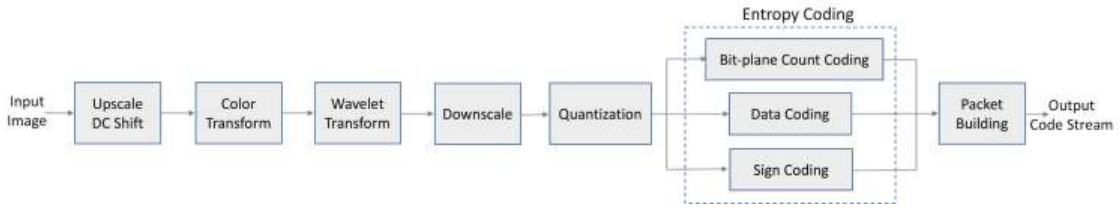


Fig. 1. Block diagram of JPEG XS encoder.

- Elimination of computationally expensive preprocessing steps such as DWT/IDWT, ensuring real-time processing feasibility.
- Maintaining compatibility with the JPEG XS standard, allowing seamless integration into existing imaging and networking workflows.
- RD performance comparable to that of conventional JPEG XS without scrambling, ensuring minimal impact on video quality.

The rest of this paper is structured as follows: Section II provides an overview of JPEG XS, while Section III details the proposed lightweight scrambled JPEG XS coding technique. Simulation results are presented in Section IV, followed by conclusions and future work in Section V.

II. JPEG XS Technical Overview

This section provides an overview of JPEG XS coding technology along with its fundamental technologies, profiles and formats.

A. Coding Technology Outline

Similar to JPEG 2000 [10], the JPEG XS core coding system is a wavelet-based still image codec. Since each frame is processed as an independent still image, JPEG XS can also function as a video codec. Fig. 1 illustrates the block diagram of the JPEG XS encoder. The encoding process begins with scaling the image data according to its bit depth, followed by DC offset removal to obtain a zero-mean signal. For RGB input, a reversible color decorrelation transformation is applied, converting it into an approximate YCbCr space - a process identical to the Reversible Color Transform (RCT) in JPEG 2000. Subsequently, a wavelet transformation is performed. The current specification supports one or two-level vertical wavelet decomposition using the LeGall 5/3 wavelet, which is also employed in JPEG 2000, and allows up to eight horizontal decomposition levels. Next, rate allocation is handled through quantization, followed by entropy coding. Finally, the encoded data is packetized to construct the JPEG XS bitstream.

B. Fundamental Technologies

1) Reversible Color Transformation: The Reversible Color Transformation (RCT) serves as a decorrelating process applied to the RGB components of an image. By eliminating the correlation between RGB components, the amount of information can be effectively reduced. The definitions of RCT and its inverse RCT in JPEG XS are expressed as follows:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} \lfloor \frac{R+2G+B}{4} \rfloor \\ R-G \\ B-G \end{bmatrix}, \tag{1}$$

$$\begin{bmatrix} G \\ R \\ B \end{bmatrix} = \begin{bmatrix} Y - \lfloor \frac{C_b+C_r}{4} \rfloor \\ C_b+G \\ C_r+G \end{bmatrix}. \tag{2}$$

2) Wavelet Decomposition: In JPEG XS, the following LeGall 5/3 wavelet transform is used.

$$y(2n+1) = x(2n+1) - \left\lfloor \frac{x(2n)+x(2n+2)}{2} \right\rfloor \tag{3}$$

$$y(2n) = x(2n) + \left\lfloor \frac{y(2n-1)+y(2n+1)+2}{4} \right\rfloor, \tag{4}$$

In this context, $y(2n+1)$ denotes the high-frequency wavelet coefficient, while $y(2n)$ corresponds to the low-frequency wavelet coefficient. Due to its compatibility with the lifting scheme, efficient processing can be achieved using simple shift operations, enabling a lightweight implementation. By applying the wavelet transform both vertically and horizontally to the input image, a two-dimensional sub-band decomposition is obtained. The resulting low-frequency subband undergoes further recursive wavelet transformations, refining the hierarchical decomposition. By this recursive processing, resolution scalability can be realized, which also contributes to the improvement of coding efficiency.

3) Quantization: JPEG XS provides both a dead-zone quantizer and a uniform quantizer. The dead-zone quantizer is implemented by removing T least significant bit (LSB) planes through truncation. When selecting a uniform quantizer, the quantization size Δ_T is set as follows:

$$\Delta_T = \frac{2^{M_g+1}}{2^{M_g+1-T}-1}. \tag{5}$$

The quantization can be easily realized only by shifting and adding. M_g is a bit plane count defined by the following Eq. (6). It represents a bit plane with significant bits.

$$M_g = \max\left(\left\lfloor \log_2 \max_{i \in g} x_i \right\rfloor + 1.0 \right), \tag{6}$$

where g represents a coding group (hereinafter, described in *4) Entropy Coding*), and x_i represents the i-th coefficient in coding group g.

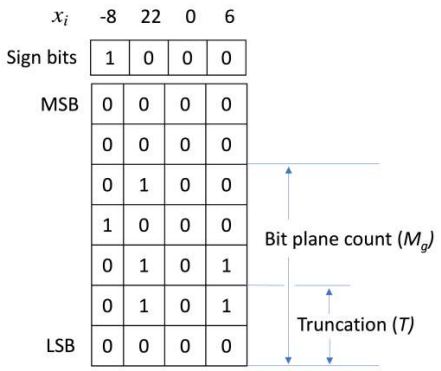


Fig. 2. Coding group g for entropy coding.

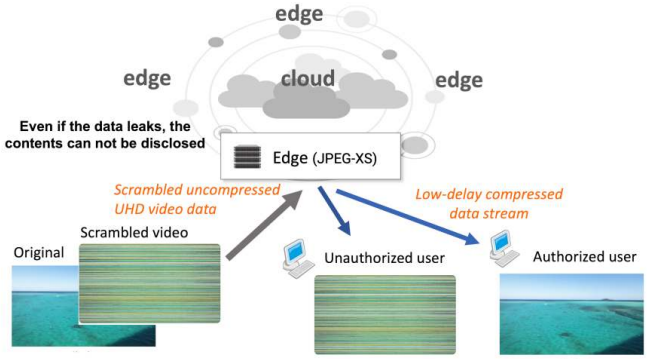


Fig. 3. The concept of scrambled JPEG XS coding.

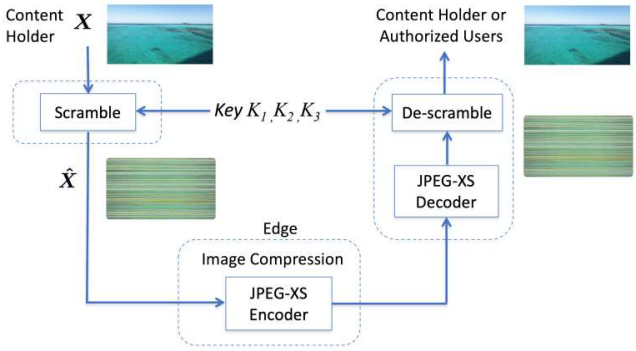


Fig. 4. The system architecture of scrambled JPEG XS.

4) Entropy Coding: In order to encode with as few bits as possible, it is common to represent frequently occurring wavelet coefficient values in short codewords and rare wavelet coefficient values in large codewords. This process is called entropy coding. Unfortunately, variable-length coding and decoding demand significant hardware and software resources. To reduce implementation complexity, JPEG XS applies variable-length coding to groups of four coefficients, referred to as coding groups, rather than encoding each wavelet coefficient individually. Fig. 2 shows an example of $x_i \in \{-8, 22, 0, 6\}$, $M_g = 5$, $T = 2$. M_g is called a "bitplane count" because it can be interpreted as the number of nonzero bitplanes in the coding group. T is a truncation point. The following processing is performed in each coding group.

1) Bit-plane count coding
   It encodes the bit plane count M_g. Several prediction modes are provided to improve coding efficiency.
2) Data coding
   It encodes the wavelet coefficient. The bit plane between the bit plane count M_g and the truncation point T is recorded in order from the MSB. In the example of Fig. 2, it is "010010000101".
3) Sign coding
   It encode the sign of the wavelet coefficient. In the example of Fig. 2, it is"1000".

C. JPEG XS Profiles and Formats

JPEG XS supports multiple profiles, including Light and Main, optimized for different use cases, from real-time streaming to high-resolution image storage. The profiles are characterized by specific parameters such as chroma subsampling (4:2:2 or 4:4:4), bit depth (10-bit or 12-bit) and wavelet decomposition levels, allowing flexibility in high-quality image transmission. By default, the main profile restricts the vertical wavelet transform to a maximum of one level. The light-subline profile achieves minimal latency and computational complexity by omitting the vertical wavelet transform entirely. In contrast, the high profile provides the highest coding efficiency at the cost of increased computational complexity, allowing for up to two levels of vertical wavelet transform.

JPEG XS defines different file and transport formats and can be used for archiving or streaming. It is based on existing standard formats such as MP4, MPEG-2 TS and RTP, allowing computer color correction rendering and video archiving or streaming.

III. The Proposed Scrambled JPEG XS

In this section, we introduce a scrambled JPEG XS coding method designed for an EtC system.

A. Design Concept and System Architecture

The concept of scrambled JPEG XS coding is illustrated in Fig. 3. This approach enhances security by preventing unauthorized access to meaningful visual information, ensuring that even if the data is intercepted, its content remains unintelligible. The proposed method maintains compatibility with standard JPEG XS compression, allowing seamless integration into existing imaging and networking workflows. This paper focuses on the design of a scrambling technique for JPEG XS, implemented as a preprocessing stage. The technique ensures: 1) compliance with the JPEG XS bitstream syntax, and 2) negligible degradation of JPEG XS's RD performance, while providing effective visual scrambling.

Fig. 4 illustrates an EtC system incorporating the proposed scrambled JPEG XS. At the local site, the input image $\boldsymbol{X}$ undergoes transformation into a scrambled image $\hat{\boldsymbol{X}}$ by applying line permutation, line reversal, and color permutation. Each operation is performed using a separate private key: K_1 for line permutation, K_2 for line reversal, and K_3 for

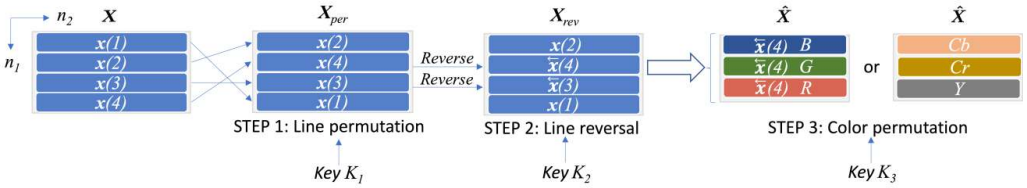


Fig. 5. An example of scrambled image generation by using line permutation, line reversal and color permutation.

color permutation. Subsequently, the scrambled image $\hat{\boldsymbol{X}}$ is transmitted to the edge or cloud. At the edge/cloud, the JPEG XS encoder compresses the scrambled image. On the receiver side, the compressed bitstream is processed by the JPEG XS decoder, resulting in a decompressed but still scrambled image. Only authorized users possessing the private keys K_1, K_2 and K_3 can successfully restore the original image by descrambling.

B. Scrambled Image Generation

Scrambled image generation consists of line permutation, line reversal, and color permutation, as shown in an example in Fig. 5. To achieve real-time UHD video compression using software or FPGA, vertical DWT is omitted, and thus, horizontal line signals are treated independently. To facilitate the description of the scrambling operations, we define an input image as follows:

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}(1) \\ \boldsymbol{x}(2) \\ \vdots \\ \boldsymbol{x}(N_1) \end{bmatrix}, \tag{7}$$

$$\boldsymbol{x}(n_1) = [x(n_1,1), x(n_1,2), \cdots, x(n_1,N_2)], \tag{8}$$

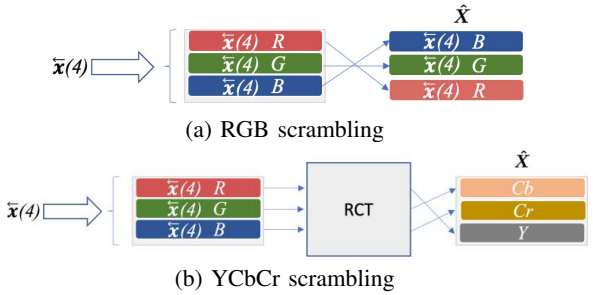
where $x(n_1,n_2)$ is the pixel value at the position $x(n_1,n_2)$, N_1 and N_2 are the number of vertical and horizontal pixels, respectively. Strictly speaking, each RGB component has its own intensity value at $x(n_1,n_2)$; however, for simplicity, the notation is omitted in this description. Image scrambling consists of two steps:

1) Line Permutation: In the initial step, the horizontal lines $\boldsymbol{x}(n_1)$ undergo random permutation using a random permutation matrix (RPM) $\boldsymbol{P}_{K_1}^{(N_1)}$ with a private key K_1. This process is formulated as follows:

$$\boldsymbol{X}_{per} = \boldsymbol{P}_{K_1}^{(N_1)} \boldsymbol{X}. \tag{9}$$

The RPM is a binary square matrix in which each row and each column contains exactly one entry of 1, with all other entries being 0. It permutates horizontal lines. An example of the line-permuted image $\boldsymbol{X}_{per}$ when $N_1 = 4$ is depicted in Fig. 5. The RPM is described by

$$\boldsymbol{P}_{K_1}^{(4)} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{10}$$



(a) RGB scrambling

(b) YCbCr scrambling

Fig. 6. Examples of color permutation.

2) Line Reversal: The second step involves reversing the order of elements within horizontal lines $\boldsymbol{x}(n_1)$. The line reversal operation is defined as:

$$\tilde{\boldsymbol{x}}(n_1) = \boldsymbol{x}(n_1)\boldsymbol{R}^{(N_2)}, \tag{11}$$

where $\boldsymbol{R}^{(N_2)} \in \{1,0\}^{N_2 \times N_2}$ denotes an anti-diagonal matrix (ADM), which is a square binary matrix containing a single entry of 1 in the reverse diagonal and 0s elsewhere. For example, for $N_2 = 4$, the ADM is given by

$$\boldsymbol{R}^{(4)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{12}$$

For example, when $N_2 = 4$ and $\boldsymbol{x}(n_1) = [1,2,3,4]$, applying horizontal reversal results in $\boldsymbol{x}(n_1)\boldsymbol{R}^{(4)} = [4,3,2,1]$. The selection of horizontal lines to be reversed is performed randomly, with the specific pattern dictated by the private key K_2. Fig. 5 presents an example where $\boldsymbol{x}(4)$ and $\boldsymbol{x}(3)$ are chosen for reversal.

3) Color Permutation: Following line permutation and line reversal, color permutation is applied to each image line. For color permutation, we propose two scrambling methods: RGB scrambling and YCbCr scrambling. Fig. 6(a) illustrates the configuration of RGB scrambling, in which the R and B components are randomly permuted. By leveraging the symmetrical properties of RCT as shown in Eq. (1), RD performance remains comparable to that without color permutation. The permutation operation is defined as follows:

$$\begin{bmatrix} \hat{R} \\ G \\ \hat{B} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{13}$$

The lines for the color permutation are randomly selected, with the chosen pattern determined by a private key K_3.

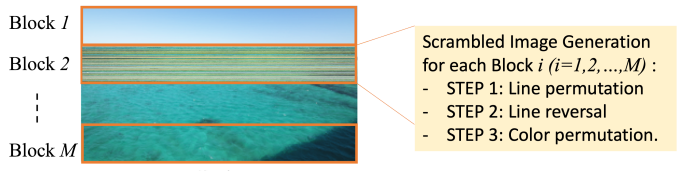


Fig. 7. Block-based image scrambling for low-latency and reduced memory usage.

Fig. 6(b) depicts the configuration of YCbCr scrambling. In this method, the RGB signals are first transformed into YCbCr components using RCT. The YCbCr components are then randomly permuted, allowing for six possible permutation patterns. An example is given below:

$$\begin{bmatrix} \hat{Y} \\ \hat{C}_b \\ \hat{C}_r \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} C_b \\ C_r \\ Y \end{bmatrix}. \tag{14}$$

RGB scrambling is fully compatible with the JPEG XS standard, as it applies scrambling before the RCT stage. In contrast, YCbCr scrambling enhances visual concealment but requires RCT as a preprocessing step.

C. Horizontally Rectangular Block Scrambling

Although processing one full frame at a time is feasible for many real-time applications, it may not be ideal when lower latency or reduced memory usage is required. In scenarios where both low latency and minimal memory usage are critical, applying scrambling to smaller horizontally rectangular blocks offers an effective solution for minimizing processing time. Unlike conventional full-frame scrambling, which requires buffering the entire frame before processing, block-based scrambling enables parallel processing of smaller image segments, significantly reducing latency. As illustrated in Fig. 7, the image is divided into M horizontally rectangular blocks, allowing the scrambling operations to be applied independently to each block. This approach enhances processing efficiency, particularly in real-time video transmission systems where immediate encoding and compression are required. Each block undergoes the following scrambling operations: 1) line permutation, 2) line reversal, and color permutation.

By employing this block-based strategy, a trade-off between security strength and processing efficiency can be achieved. A higher M value results in smaller block sizes, reducing latency and memory usage, but it may weaken scrambling strength due to increased spatial correlation within blocks. Conversely, a lower M value enhances security by increasing randomness, but at the cost of higher processing overhead. This flexibility allows the method to be adapted for various real-time applications, including low-latency UHD video streaming and edge/cloud-based video processing.

D. Security Strength

We assessed the security strength of the spatio-color scrambled image $\hat{\boldsymbol{X}}$ with regard to the key spaces associated with line permutation, line reversal, and color permutation. The key space is evaluated under the assumption of restoration via a brute-force attack. Initially, we analyze the key space associated with line permutation $\boldsymbol{P}_{K_1}^{(N_1)}$. Authorized users possessing the private key K_1 are able to reconstruct the original input image by

$$\boldsymbol{X} = [\boldsymbol{P}_{K_1}^{(N_1)}]^{-1} \boldsymbol{X}_{per}. \tag{15}$$

The RPM satisfies the property $[\boldsymbol{P}_{K_1}^{(N_1)}]^{-1} = [\boldsymbol{P}_{K_1}^{(N_1)}]^T$, where $[*]^{-1}$ denotes the inverse operation, and $[*]^T$ represents the transpose operation. The key space associated with $\boldsymbol{P}_{K_1}^{(N_1)}$ is determined by $N_1!$, as it solely depends on the number of vertical pixels. Next, we examine the key space for line reversal. Each horizontal line can be arranged in two possible states: either reversed or maintained in its original order. With N_1 rows, there exist 2^{N_1} combinations. Therefore, the key space for line reversal is 2^{N_1}. Finally, we explore the key space for color permutation. In RGB scrambling, each horizontal line exhibits two patterns: swapping the R and B components or retaining their order. With N_1rows, this results in 2^{N_1} combinations. For YCbCR scrambling, each horizontal line has six patterns, yielding a key space of 6^{N_1}. In summary, the key spaces of the scrambled images for RGB scrambling and YCbCr scrambling are shown in Table I(a).

Finally, we will look at the key space in block-based image scrambling. The combination pattern per block is $(N_1/M)! \times 2^{(N_1/M)} \times 2^{(N_1/M)}$ for RGB scrambling and $(N_1/M)! \times 2^{(N_1/M)} \times 6^{(N_1/M)}$ for YCbCr scrambling. When $M = 1$, the system operates in the *non-block* mode, meaning

TABLE I. KEY SPACES OF SCRAMBLED IMAGES

(a) *Non-block* scrambling

Non-block RGB	$N_1! \times 2^{N_1} \times 2^{N_1}$
Non-block YCbCr	$N_1! \times 2^{N_1} \times 6^{N_1}$

(b) Block-based scrambling

Block RGB	$\{(N_1/M)! \times 2^{(N_1/M)} \times 2^{(N_1/M)}\}^M$
Block YCbCr	$\{(N_1/M)! \times 2^{(N_1/M)} \times 6^{(N_1/M)}\}^M$

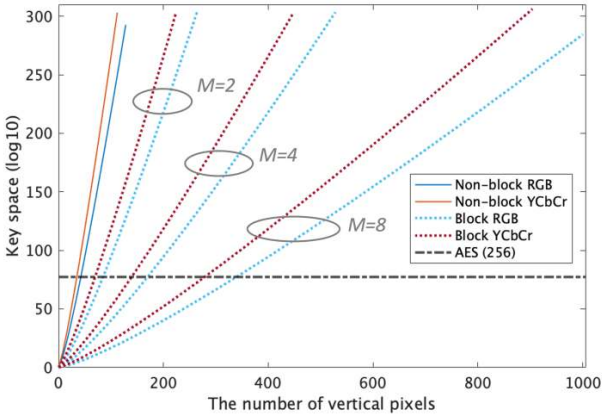
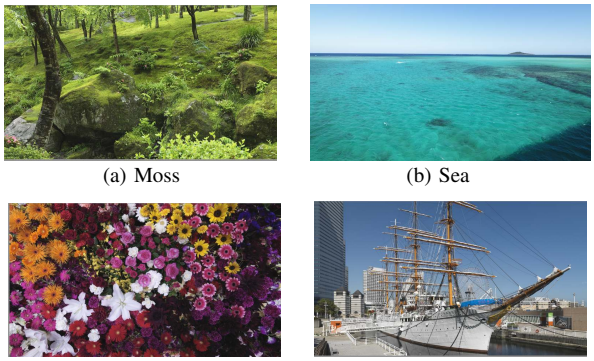


Fig. 8. The key space of the *non-block* scrambled images, and block based scrambled images for different values of M ($M = 2, 4$, and 8).



(a) Moss (b) Sea

(c) Flowers (d) Ship

Fig. 9. Original 8K images [32].

TABLE II. MEAN ABSOLUTE PEARSON PRODUCT-MOMENT CORRELATION COEFFICIENT (PPMC) BETWEEN ORIGINAL 8K IMAGES AND THE CORRESPONDING DESCRAMBLED 8K IMAGES

(a) RGB Scrambling

M	*Non-block*	2	4	8
Moss	0.0259	0.0444	0.0917	0.149
Sea	0.0041	0.476	0.561	0.645
Flowers	0.0182	0.0243	0.0666	0.0954
Ship	0.0264	0.0683	0.0994	0.1424

(b) YCbCr Scrambling

M	*Non-block*	2	4	8
Moss	0.0180	0.0339	0.0659	0.106
Sea	0.0102	0.243	0.289	0.332
Flowers	0.0140	0.0208	0.0517	0.0752
Ship	0.0162	0.0181	0.0304	0.0531

that no division into multiple blocks. Table I presents a comprehensive summary of our key space analysis results.

Fig. 8 illustrates the calculated key space for both *non-block* and block-based scrambled images with varying block sizes M, based on the formula in Table I. The results indicate that a larger block size corresponds to a smaller key space. For comparison, the key space of 256-bit AES is also included. In the case of UHD images, with vertical resolutions typically exceeding 2000 pixels, the key spaces for both *non-block* and block-based scrambled images are sufficiently large.

IV. EXPERIMENTAL RESULTS

We processed four 8K images, namely "Moss", "Sea", "Flowers", and "Ship", shown in Fig. 9, obtained from the Institute of Image Information and Television Engineers (ITE) [32]. These images possess a resolution of $N_1 = 4320$, $N_2 = 7680$ with a depth of 36 bits (12 bits/color), adhering to the UHDTV studio standard Recommendation ITU-R BT.2020 (Rec. 2020) [33].

A. Visibility of Scrambled Images

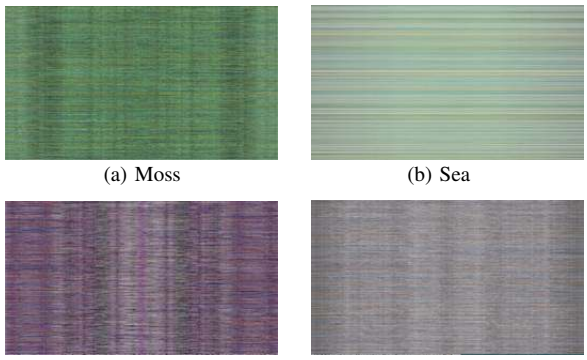
The effectiveness of the proposed scrambling method is evaluated based on visual obscuration, where a higher level of distortion indicates stronger scrambling performance. Scrambled images generated using the proposed spatio-color scrambling method are shown in Fig. 10-11, confirming its strong visual concealment capability. Among the methods, YCbCr scrambling demonstrates the highest level of invisibility. In particular, for the "Sea" and "Ship" images, the original content is nearly unrecognizable. As an example, Fig. 13 presents the frequency distribution of RGB values in the "Sea" image. The proposed method shows significantly reduced color bias compared to the original image, which exhibits strong skewness in RGB component distribution. This effect is especially pronounced in YCbCr scrambling, where the RGB distribution appears nearly uniform.

Fig. 12 shows the "Ship" images after applying the rectangular block-based RGB scrambling method with block division parameters M= 2, 4, and 8. For comparison, the non-block version is also included. As M increases, the scrambling performance decreases, making the original image slightly more visible. This degradation is due to smaller block sizes preserving local spatial correlations, which reduces visual distortion. In contrast, a lower M yields stronger scrambling effects, effectively obscuring the original content. Although higher M values weaken scrambling strength, they help reduce transmission latency and memory usage by limiting the number of blocks to be processed. This trade-off should be carefully considered according to application requirements. These results indicate that the choice of M significantly influences scrambling effectiveness. Lower values of M are preferable when higher security and stronger image concealment are needed, while higher values may be more suitable for applications prioritizing low latency and reduced memory, despite a slight decrease in scrambling strength.

B. RD Performance

The efficacy of the proposed spatio-color scrambled JPEG XS scheme, using RGB scrambling, was evaluated in terms of RD performance. A comparative analysis was conducted against the non-scrambled version of JPEG XS. Fig. 14 illustrates the RD performance of both methods: the solid line represents the non-scrambled JPEG XS, while the dotted line represents the proposed method. In the proposed scheme, PSNR is calculated by comparing the images decoded by an authorized user with the original images. Compared to the non-scrambled version, the proposed scheme exhibits only marginal degradation in RD performance, while simultaneously improving invisibility, as shown in Fig. 10. At higher bit rates, the PSNR difference becomes slightly more noticeable, but it remains above 40 [dB] - within a range generally imperceptible to the human eye.

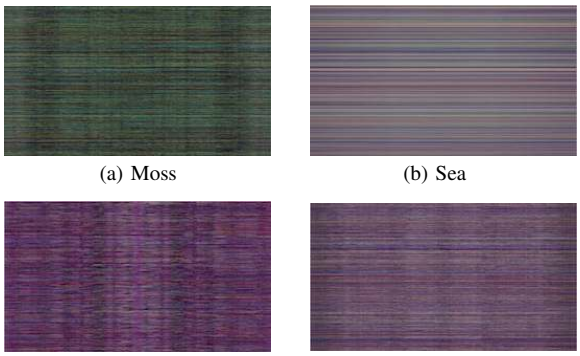
Fig. 15 illustrates the decoded "Ship" images with RGB scrambling at bitrates of 2 [bpp] and 10 [bpp], as viewed by both authorized and unauthorized users. An authorized user possessing the correct private keys can successfully decode the scrambled images, whereas an unauthorized user fails to do so. Fig. 16 presents partially enlarged views of the decoded "Ship" images at the same bitrates. The images labeled "Authorized user" represent the decoded results for an authorized user, while those labeled "Non-scrambled JPEG XS" correspond to the decoded images produced without scrambling. These figures demonstrate that the visual characteristics of the decoded images remain nearly identical, regardless of whether scrambling was applied.



(a) Moss (b) Sea

(c) Flowers (d) Ship

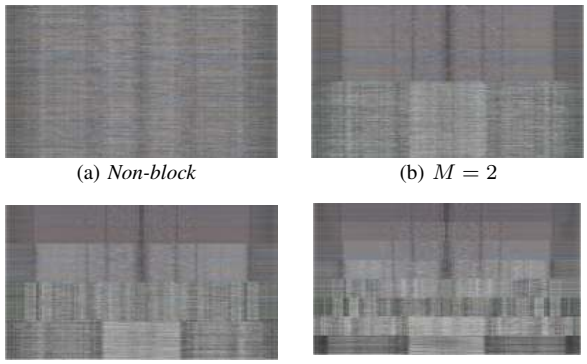
Fig. 10. The proposed spatio-color scrambled 8K images using RGB scrambling.



(a) Moss (b) Sea

(c) Flowers (d) Ship

Fig. 11. The proposed spatio-color scrambled 8K images using YCbCr scrambling.



(a) *Non-block* (b) $M = 2$

(c) $M = 4$ (d) $M = 8$

Fig. 12. The proposed block-based spatio-color scrambled 8K images using RGB scrambling for "Ship" image.

C. Security Strength

We assessed the security robustness of $\hat{\boldsymbol{X}}$ under the assumption of restoration via brute-force attack. The Pearson product-moment correlation coefficient (PPMC) and mean squared error (MSE) were used as similarity metrics. A lower PPMC and higher MSE indicate stronger scrambling, while the opposite implies weaker scrambling. Two samples are typically considered uncorrelated when the absolute value of their PPMC is below 0.2; values approaching 1 indicate strong correlation. To simulate unauthorized access, 100 random descrambling patterns were generated. Table II shows the mean absolute PPMC values obtained over 100 trials, comparing original 8K images with their descrambled counterparts using both RGB and YCbCr scrambling methods. Table III presents the corresponding MSE values. For clarity, all values are expressed in units of 10^8.

TABLE III. MEAN SQUARED ERROR (MSE) $[\times 10^8]$ BETWEEN ORIGINAL 8K IMAGES AND THE CORRESPONDING DESCRAMBLED 8K IMAGES

(a) RGB Scrambling

M	*Non-block*	2	4	8
Moss	3.27	3.22	3.07	2.90
Sea	7.51	4.55	4.04	3.51
Flowers	6.15	6.11	5.86	5.69
Ship	3.93	3.74	3.64	3.46

(b) YCbCr Scrambling

M	*Non-block*	2	4	8
Moss	4.64	4.58	4.45	4.30
Sea	7.29	5.57	5.25	4.95
Flowers	6.27	6.23	6.06	5.92
Ship	5.16	5.15	5.09	4.98

From Tables II and III, the non-block method exhibits strong scrambling performance, as its mean absolute PPMC values remain consistently low for both RGB and YCbCr scrambling. Furthermore, within the non-block method, YCbCr scrambling generally demonstrates stronger scrambling performance than RGB scrambling, except for the "Sea" image. In this case, which includes a high proportion of blue components, RGB scrambling - particularly the swapping of the R and B channels - appears to improve scrambling strength. This suggests that image color characteristics can influence the effectiveness of different scrambling methods.

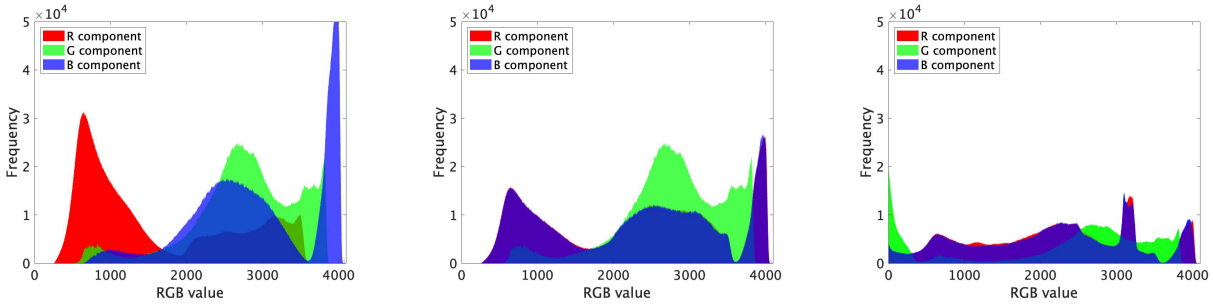
For rectangular block-based scrambling (with $M \geq 2$), increasing M tends to raise the mean absolute PPMC and lower the MSE, indicating weakened scrambling strength. In the case of the "Sea" image, the mean absolute PPMC values exceed 0.2 under block-based scrambling, likely because the upper blocks contain sky and the lower blocks contain sea, resulting in similar structures within each block. Consequently, line permutation alone cannot sufficiently disrupt these spatial correlations. Therefore, while block-based scrambling offers flexibility in balancing security and computational efficiency, its effectiveness must be carefully evaluated, particularly for large M values or images with distinct regional segmentation.

V. DISCUSSION

The proposed lightweight scrambled JPEG XS coding scheme has demonstrated its effectiveness in maintaining high visual privacy while ensuring compatibility with standard JPEG XS compression. In this section, we discuss key findings, potential limitations regarding the proposed approach.

A. Impact on Rate-Distortion Performance

Our experimental results confirm that the proposed scrambling technique introduces minimal degradation in RD performance. The RD curves indicate that the PSNR values of the decoded images remain comparable to those of conventional JPEG XS, even when scrambling is applied. This suggests



(a) Original RGB (b) RGB scrambling (c) YCbCr scrambling

Fig. 13. Frequency distribution of RGB color space in "Sea" image.

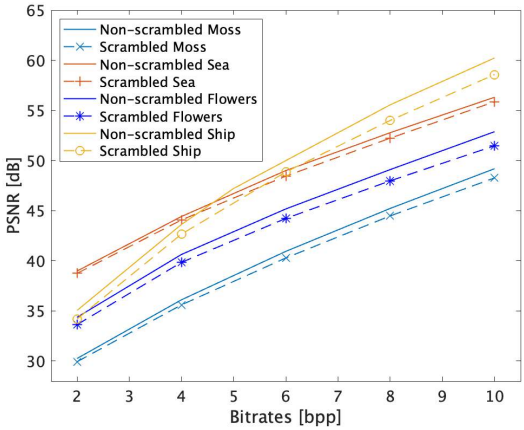


Fig. 14. Rate-distortion performance of the proposed scrambled JPEG XS and the non-scrambled JPEG XS for 8K images.

Authorized user Unauthorized user

(a) Bitrates = 2 [bpp]

Authorized user Unauthorized user

(a) Bitrates = 10 [bpp]

Fig. 15. Decoded images of the RGB scrambled "Ship" images at bitrates of 2 [bpp] and 10 [bpp].

Authorized user Non-scrambled JPEG XS

(a) Bitrates = 2 [bpp]

Authorized user Non-scrambled JPEG XS

(a) Bitrates = 10 [bpp]

Fig. 16. Partially enlarged areas of the decoded "Ship" images at bitrates of 2 [bpp] and 10 [bpp].

that the proposed method effectively preserves visual quality while providing privacy protection. At higher bit rates, a slight decrease in PSNR can be observed due to the applied scrambling. However, the PSNR consistently remains above 40 dB, a level at which differences are generally imperceptible to the human eye. Therefore, the impact on perceived image quality is negligible.

B. Security Considerations and Trade-offs

The security analysis confirms that the proposed scrambling scheme significantly expands the key space, making brute-force attacks infeasible. The combination of line permutation, line reversal, and color permutation effectively prevents unauthorized reconstruction. In the case of the block-based approach, the scrambling strength depends on the number of block divisions M. A higher M reduces latency and memory but may weaken security due to increased spatial correlation within blocks. Conversely, a lower M enhances security by increasing randomness but at the cost of higher processing overhead. This trade-off must be carefully balanced based on application needs.

C. Applicability to Real-time UHD Video Transmission

One of the primary advantages of the proposed method is its suitability for real-time UHD video transmission. Unlike conventional EtC schemes based on block-based image coding (e.g., JPEG or JPEG2000), our approach is optimized for the JPEG XS framework, ensuring lightweight and low-latency processing. The proposed scheme can be seamlessly integrated into existing JPEG XS-based imaging pipelines, making it practical for deployment in B5G applications such as edge/cloud-based video processing.

D. Limitations

While the proposed method provides lightweight processing and high visual security, there are two primary limitations. One key limitation is that, in prioritizing low latency and low computational complexity, the method omits vertical DWT. As a result, its RD performance is inferior to that of schemes applying both vertical and horizontal DWT. Another limitation arises from the block-based scrambling approach: as the number of block divisions M increases, scrambling strength tends to decline due to the preservation of local spatial structures within smaller blocks, potentially reducing visual concealment.

VI. Conclusions and Future Work

This paper presented a lightweight EtC scheme for the JPEG XS standard, incorporating line permutation, line reversal, and color permutation to scramble input images prior to compression. The proposed approach is compatible with JPEG XS and is designed to enhance visual privacy. Extensive simulations using 8K UHD images demonstrated that the scrambling technique achieves RD performance nearly equivalent to conventional JPEG XS compression. Moreover, it improves visual concealment, with subjective evaluations indicating that the scrambled images effectively obscure meaningful content from unauthorized viewers. The block-based variant contributes to reduced latency and memory usage while offering a reasonable trade-off in scrambling strength, depending on the block division parameter M.

Future work will focus on enhancing the security of the block-based scrambling scheme. In addition, optimization techniques for both software and FPGA implementations should be explored to improve latency, computational efficiency, and memory usage. While the current evaluation is limited to still 8K images, extending the method to video sequences is also necessary to assess temporal consistency and reduce potential motion artifacts. Finally, developing adaptive scrambling techniques that dynamically adjust security levels based on network conditions and application requirements could further improve the flexibility and robustness of the proposed approach.

Acknowledgment

This work is partly supported by the commissioned research JPJ012368C03101 by National Institute of Information and Communications Technology (NICT) Japan, and JST CRONOS Japan Grant Number JPMJCS24N9.

References

[1] M. Maruyama, et al., "Ultra-high-speed in-network computing platform", JST CRONOS Japan.

[2] H. Kimiyama et al., "Proposal of ultra-high-resolution video delivery system in edge-cloud environment," 2022 IEEE International Conference on Consumer Electronics - Taipei, Taiwan, 2022, pp. 331-332, doi: 10.1109/ICCE-Taiwan55306.2022.9869110.

[3] K. Sebayashi, et al., "Uncompressed 8K video processing using SRv6-based service function chaining between Japan and the U.S," The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC23), Network Research Exhibition, 2023.

[4] JPEG XS Low-latency lightweight image coding system - Part 1: core coding cystem, Standard ISO/IEC 21122-1:2019, 2019.

[5] JPEG XS Low-latency lightweight image coding system - Part 2: profiles and buffer models, standard ISO/IEC 21122-2:2019, 2019.

[6] JPEG XS low-latency lightweight image coding system - Part 3: transport and container formats, Standard ISO/IEC 21122-3:2019, 2019.

[7] JPEG White paper: JPEG XS, a new standard for visually lossless low-latency lightweight image coding system, ISO/IEC JT1/SC29/WG1 WG1N83038.

[8] A. Descampe et al., "JPEG XS - A new standard for visually lossless low-latency lightweight image coding," in Proceedings of the IEEE, vol. 109, no. 9, pp. 1559-1577, Sept. 2021, doi: 10.1109/JPROC.2021.3080916.

[9] Use cases and requirements for ISO/IEC 21122-1 (JPEG XS Part-1, core coding system) v2.1, standard ISO/IEC JTC 1/SC 29/WG1, Oct. 2020.

[10] Information technology -JPEG2000 image coding system: core coding system, ISO/IEC 15444-1:2004 — ITU-T Rec. T.800. 2015.

[11] C.T. Huang, et al., "Survey on securing data storage in the cloud," APSIPA Transactions on Signal and Information Processing, vol.3, e7, 2014.

[12] J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu, "Data security and privacy-preserving in edge computing paradigm: Survey and open issues," IEEE Access, vol. 6, pp. 18209-18237, 2018.

[13] A. Mishra, T. S. Jabar, Y. I. Alzoubi, K. N. Mishra, "Enhancing privacy-preserving mechanisms in Cloud storage: A novel conceptual framework," Concurrency and Computation: Practice and Experience, vol. 35, no. 10, June 2023.

[14] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," IEEE Transactions on Signal Processing, vol. 52, no. 10, pp. 2992-3006, 2004.

[15] D. Schonberg, S. C. Draper, C. Yeo, and K. Ramchandran, "Toward compression of encrypted images and video sequences," IEEE Transactions on Information Forensics & Security, vol. 3, no. 4, pp. 749-762, 2008.

[16] W. Liu, W. Zeng, L. Dong, and Q. Yao, "Efficient compression of encrypted grayscale images," IEEE Transactions on Image Processing, vol. 19, no. 4, pp. 1097-1102, 2010.

[17] X. Zhang, "Lossy compression and iterative reconstruction for encrypted image," IEEE Transactions on Information Forensics & Security, vol. 6, no. 1, pp. 53-58, 2011.

[18] J. Zhou, X. Liu, O. C. Au, and Y. Y. Tang, "Designing an efficient image encryption-then-compression system via prediction error clustering and random permutation," IEEE Transactions on Information Forensics & Security, vol. 9, no. 1, pp. 39-50, 2014.

[19] C. Wang, J. Ni, and Q. Huang, "A new encryption-then-compression algorithm using the rate-distortion optimization," Signal Processing: Image Communication, vol. 39, pp. 141-150, 2015.

[20] M. Kumar and A. Vaish, "An efficient encryption-then-compression technique for encrypted images using SVD," Digital Signal Processing, vol. 60, pp. 81-89, 2017.

[21] K. Kurihara, M. Kikuchi, S. Imaizumi, S. Shiota, and H. Kiya, "An encryption-then-compression system for JPEG/motion JPEG standard," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. 98-A, no. 11, pp. 2238-2245, 2015.

[22] O. Watanabe, A. Uchida, T. Fukuhara, and H. Kiya, "An encryption-then-compression system for JPEG 2000 standard," 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, pp. 1226-1230, 2015.

[23] K. Kurihara, S. Imaizumi, S. Shiota, and H. Kiya, "An encryption-then-compression system for lossless image compression standards," IEICE Transactions on Information and Systems, vol. 100-D, no. 1, pp. 52-56, 2017.

[24] S. Imaizumi and H. Kiya, "A block-permutation-based encryption scheme with independent processing of RGB components," IEICE Transactions on Information and Systems, vol. E101.D, no. 12, pp. 3150-3157, 2018.

[25] T. Chuman, K. Iida, W. Sirichotedumrong, and H. Kiya, "Image manipulation specifications on social networking services for encryption-then-compression systems," IEICE Trans. Inf. & Syst., vol.E102.D, no.1, pp.11-18. Jan. 2019.

[26] T. Chuman, W. Sirichotedumrong, and H. Kiya, "Encryption-then-compression systems using grayscale-based image encryption for JPEG images," IEEE Trans. Inf. Forensics Security, vol.14, no.6, pp.1515-1525, June 2019.

[27] T. Nakachi, H. Kiya, "Secure OMP computation maintaining sparse representations and its application to EtC systems," IEICE Transactions on Information and Systems, vol. E103-D, no. 9, pp. 1988-1997, 2020.

[28] T. Nakachi, Y. Bandoh, H. Kiya, "Secure overcomplete dictionary learning for sparse representation," IEICE Transactions on Information and Systems, vol. E103.D, no. 1, pp. 50-58, 2020.

[29] C. Li, S. Liu, "Recovering the block-wise relationship in an encryption-then-compression system," arXiv:2305.04543, May 2023.

[30] T. Nakachi, H. Kimiyama and M. Maruyama, "Lightweight scrambled JPEG XS coding for privacy protection," 2022 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Penang, Malaysia, 2022, pp. 1-4, doi: 10.1109/ISPACS57703.2022.10082851.

[31] T. Nakachi, H. Kimiyama and M. Maruyama, "A lightweight spatio-color scrambled EtC system for JPEG XS standard," 2024 7th International Conference on Information and Computer Technologies (ICICT), Honolulu, HI, USA, 2024, pp. 228-232, doi: 10.1109/ICICT62343.2024.00042.

[32] ITE, "Ultra-High Definition/Wide-Color-Gamut Standard Test Images," https://www.ite.or.jp/content/chart/uhdtv/, 2014

[33] Rec. ITU-R BT.2020, "Parameter values for ultra-high definition television systems for production and international programme exchange," Aug. 2012.