applied sciences

MDPI

Article

XtoE: A Novel Constructive and Camouflaged Adaptive Data Hiding and Image Encryption Scheme for High Dynamic Range Images

Chi-Feng Lan, Chung-Ming Wang * and Woei Lin

Department of Computer Science and Engineering, National Chung Hsing University, Taichung 402, Taiwan
* Correspondence: cmwang@cs.nchu.edu.tw

Abstract: High dynamic range (HDR) image data hiding and encryption has attracted much interest in recent years due the benefits of providing high quality realistic images and versatile applications, such as copyright protection, data integrity, and covert communication. In this paper, we propose a novel constructive and camouflaged adaptive data hiding and image encryption scheme for HDR images. Our algorithm disguises hidden messages when converting an original OpenEXR format to the RGBE encoding, which contains the Red, Green, and Blue color channels and an exponent E channel. During the conversion process, we determine an optimal base for each pixel by considering the user's demands and the exponent E channel information to achieve adaptive message concealment. To prevent inappropriate access to the stego image, we perform the bit-level permutation and confusion using a 2D Sine Logistic modulation map with hyperchaotic behavior and a random permutation scheme with the time complexity of $O(N)$. To the best of our knowledge, our algorithm is the first in HDR data hiding literature able to predict the image distortion and satisfy a user's request for the embedding capacity. Our algorithm offers 18% to 32% larger embedding rate than that provided by the current state-of-the-art works without degrading the quality of the stego image. Experimental results confirm that our scheme provides high security superior to the competitors.

Keywords: high dynamic range image; adaptive data hiding; image encryption; prediction; security evaluation

Citation: Lan, C.-F.; Wang, C.-M.; Lin, W. XtoE: A Novel Constructive and Camouflaged Adaptive Data Hiding and Image Encryption Scheme for High Dynamic Range Images. *Appl. Sci.* **2022**, *12*, 12856. https://doi.org/10.3390/app122412856

Academic Editors: David Megías, Minoru Kuribayashi and Wojciech Mazurczyk

Received: 4 November 2022
Accepted: 12 December 2022
Published: 14 December 2022

1. Introduction

High dynamic range (HDR) images [1,2] have attracted much interest in recent years because they offer a more dynamic range, allowing displays to show images in a clearer way, especially in the details of the highlights and shadows. Highlights can be shown brighter and truly dark, and deep shadows can be displayed. Thus, the original image can be shown more completely and faithfully, closer to what the human eye can see and perceive. Dynamic range is the ratio between the brightest and darkest values that a display can show. Using the floating-point number to store crucial information, HDR images can faithfully represent a large range of luminance and colors, thereby offering great potential to become the leading image standard.

There are three main HDR image encoding formats: RGBE [3], LogLuv [4], and OpenEXR [5]. Any two formats can be converted in either a forward or backward manner. Tone-mapping [1,2,6], an essential step for HDR images, consists of adjusting the tonal values of an image with a high dynamic range so that it can be viewed on digital displays. Tone-mapping operator, TMO, scales down the dynamic range while attempting to preserve the appearance of the original image. It is thus applied to HDR images to reveal their full details and give them a dynamic twist and realistic look, providing the best image presentation for visualizing an HDR image.

Data hiding [7], also known as steganography, is a technique developed to embed secret messages into digital media. The camouflaged media looks like the original harmless one, thus arousing no suspicion among malicious eavesdroppers. The media before

message embedding is called a cover, and the media after the message concealment is referred to as a stego. In the era of internet transmission and cloud computing, data hiding provides a method for communicating secretly. While time-varying media, such as video, can be used as a carrier [8], static images are the most popular because they are widely transmitted over the Internet. Data hiding schemes using conventional low dynamic range (LDR) images, such as binary, grayscale, color, or palette images, have been presented in the extant literature [7,9,10].

Despite HDR images having the potential to become a leading standard, limited numbers of HDR data hiding works have been proposed in the literature. Similar to LDR image data hiding, the number of bits that an HDR image can carry is called the embedding capacity (EC). Taking the image resolutions into consideration, the embedding rate (ER) represents the number of bits concealed in a pixel, also denoted as bits per pixel (bpp). One of the goals pursued by a data hiding algorithm is providing an embedding rate that is large enough for practical applications. Cheng and Wang [11] proposed an adaptive data hiding and authentication algorithm for HDR RGBE images. They provided an embedding rate in the range of 5.13 to 9.69 bpp. The peak signal-to-noise ratios (PSNR) of the tone-mapped stego images are only slightly greater than the 30 dB. Li et al. [12] adopted HDR images with a 48-bit TIFF format and improved the embedding rate of Cheng and Wang's scheme. Their algorithm offered an average embedding rate of 26 bpp, with the PSNR of the tone-mapped stego images between 30.47 and 37.00 dB. Chang et al. [13], Wang et al. [14], and Yu et al. [15] took advantage of modifying a homogeneity index to achieve distortion-free data hiding in HDR RGBE images. However, the penalty of this approach limited the embedding rate to only 0.127–0.145 bpp. Lin et al. [16] presented a novel data hiding algorithm (DHOB) and an aggressive bit encoding and decomposition (ABED) scheme for OpenEXR images. Depending on the parameters, the embedding rate was between 2.433 and 20.002 bpp. He et al. [17] and Gao et al. [18] employed prediction error expansion to embed a secret message in every individual Red, Green, and Blue color channel. Their algorithms provided embedding rates ranging from 1.202 to 2.85 bpp. Tsai et al. [19] applied the multiple-base notational system and homogeneity index modification and proposed an authentication algorithm for RGBE images. With the aid of the distortion tolerance, the embedding rate reached 2.27 bpp. Finally, Tsai et al. [20] extended the multi-MSB prediction and Huffman coding developed for LDR image and proposed reversible data hiding in encrypted RGBE images. Their scheme offered an embedding rate of 6.19–7.03 bpp.

Adaptive message embedding is an effective way for hiding data in LDR or HDR images. Since an HDR image can represent a large range of luminance and human eyes are not sensitive to the dark luminance, the darker pixels are better candidates than the brighter ones to carry more messages. Cheng and Wang's algorithm [11] classified pixels of RGBE images into flat or boundary areas to embed different quantities of secret messages. Yu et al. [15] used the homogeneous representations inherent in the RGBE pixels to conceal different numbers of messages. Gao et al. [18] employed a two-level pixel prediction mechanism in complex regions of an RGBE image to reduce the distortion caused by message embedding. Targeting the OpenEXR images, Lin et al. [16] divided pixels into three categories based on the luminance and embedded more messages in low luminance pixels.

In the above works, secret messages were embedded by modifying pixels in a cover HDR image in order to generate a stego pixel, where the cover image and the stego image share the same image format. Data hiding using pixel modification inevitably produces interference with the natural features of a cover image, even though the modification is minor and even subtle. Consequently, the more that the information is concealed, the greater the image reveals distortion, which makes it difficult for these classical data hiding schemes to resist the attack of steganalysis, which intends to detect any secret messages hidden in an innocuous image. To resolve the potential steganalytic risk, a new data hiding scheme has been designed, which conceals secret messages by directly constructing a stego image rather than modifying the existing covers. This approach is termed "constructive

data hiding" or "constructive steganography". Technical algorithms developed to construct a stego image include texture synthesis [21] and color transfer [22], among others.

Image encryption [23,24] is an effective way for preserving privacy and maintaining security. While a number of low dynamic range (LDR) image encryption schemes have been presented [25,26], and unfortunately, research works on HDR encryption are limited. Yan et al. [27] used the elementary cellular automata (ECA) as an encryption tool to cipher RGBE images. Lin et al. [28] used a logistic map to generate a pseudo-random number sequence in order to encrypt an HDR image with the LogLuv format. Chen and Chang [29] encrypted the bits in the exponent field in OpenEXR images and obtain good encryption performance. Chen and Yan [30] proposed an encryption and authentication scheme for OpenEXR images; they made use of the torus automorphism as the permutation function and Vernam cipher as the stream cipher, where the plaintext is combined with a pseudo-random stream of data to generate the ciphertext using the Boolean "exclusive or" (XOR) function. Finally, Tsai et al. [20] reserved room for data hiding in the RGBE image, and then encrypted the image using a series of random binary digits. Their scheme is one of few algorithms in the extant literature that can conceal a secret message and encrypt an HDR image.

Security is perhaps the most important issue for image encryption [31,32]. However, the current state-of-the-art HDR encryption algorithms fail to analyze the security comprehensively. Some algorithms only present visual perception metric without further analysis, for example, Lin et al. [28] and Tsai et al. [20]. Apart from visual perception metric, some reported extra metrics including histograms and pixel correlation/key security results; for example, Yan et al. [27], Chen and Change [29], and Chen and Yan [30], which represent the most comprehensive work for security analysis. To the best of our knowledge, entropy analysis and image sensitivity analysis have never been presented by these HDR encryption algorithms in the extant literature. Without thorough security analysis, it remains unclear whether the image cipher actually does have a high security level able to resist malicious attacks, such as entropy attacks and differential attack [23,24].

This paper proposes a novel constructive data hiding and chaotic sequence-based bit-level image encryption algorithm. Our scheme offers six significant features. First, we conceal secret messages when converting an HDR image from the OpenEXR format into the RGBE one. From the steganalysis point of view, our scheme is able to resist the steganalytic attack because we construct a stego image rather than generate it from the existing cover image. Second, we adaptively embed a secret message according to the distribution of the exponent channel shared by the Red, Green, and Blue channels. The adaptivity enables our scheme to conceal more secret messages in pixels with low luminance, where their exponent value is smaller than the median of the whole HDR image. Third, we exploit adaptive message embedding to satisfy a user's request. A user can request a large embedding rate in exchange for endurable image quality. Alternatively, an end user can exchange a limited embedding rate for a high quality stego image. Fourth, we introduce a chaotic sequence-based bit-level image encryption algorithm which not only protects the image contents, but also secures the hidden messages, thereby reducing the probability of any inappropriate use of the stego HDR RGBE image produced. We adopt a 2D Sine Logistic modulation map (2D-SLMM) [33] to generate pseudo-random sequences. The map holds two large positive Lyapunov exponent (LE) values and two large Kolmogorov entropy values, thereby providing better hyperchaotic behavior, more complexity, and unpredictability. Fifth, we apply the random permutation technique to our scheme in a bit-level permutation mechanism. As evidence from the experimental results, the bit-level permutation significantly increases the security of HDR image encryption. The final feature of our scheme is that we present six metrics to thoroughly and comprehensively evaluate the security of the cipher image.

The main contributions of our work can be summarized as follows:

- We present a novel adaptive data hiding algorithm for an HDR image. Secret messages are concealed during the HDR format conversion, signifying that the cover

and stego image are in different image formats, avoiding the vulnerability caused by conventional data hiding utilizing the cover image modification approach.

- Our algorithm adaptively embeds secret messages. More secret messages are conveyed in those pixels containing low luminance appearing to be dark pixels, while fewer secret messages are injected into pixels with high luminance which appear to be brighter; this coincides with the phenomena that human eyes are less sensitive to the dark pixels and more sensitive to noticing subtle changes encountered in the brighter area.
- Our data hiding scheme provides the ability to satisfy the request for the embedding rate proposed by the end user. In addition, image distortion due to the message concealment can be predicted prior to real message embedding. Thanks to the optimal base mechanism we introduce, which not only satisfies users' demand, but also derives the minimal mean squared error to reduce the image distortion as small as possible. To the best of our knowledge, no algorithm in the HDR data hiding literature can comply with the embedding rate requested by the end user. In addition, there is no algorithm in the HDR imaging literature which can predict the distortion in terms of the mean squared error caused by the message concealment.
- We present a secure image encryption scheme, which adopts a 2D-SLMM to generate a pseudo-random sequence with better hyperchaotic behavior. Pixels are ciphered through the bit-level permutation using the random permutation algorithm with the time complexity of $O(N)$. As a result, our scheme can completely shuffle the whole bits aligned by the Red, Green, Blue and Exponent channel in an HDR image.
- We present a thorough and comprehensive security analysis for stego HDR RGBE images. Our scheme offers 18% to 32% more embedding rate than that provided by the current state-of-the-art works without impairing the quality of stego image. In addition, our scheme provides high image encryption security, surpassing those of our competitors. To the best of our knowledge, our scheme is the first that can report the security of HDR cipher image from six aspects: visual perception, histogram, correlation, entropy, image sensitivity, and key security.

The rest of this paper is organized as follows: Section 2 presents our proposed methods in detail. The experiments and analysis are given in Section 3. Finally, Section 4 describes conclusions and future work.

2. Our Proposed Methods

This section details our proposed algorithm for adaptive data hiding and image encryption for HDR images. Figure 1 exhibits the flowchart of our algorithm. In the sender part, two main processes are concurrent image conversion and adaptive message embedding (CICAME) and image encryption, while in the recipient part, there are image decryption and message extraction processes. Here, we briefly highlight these processes and will detail them in the following subsections.

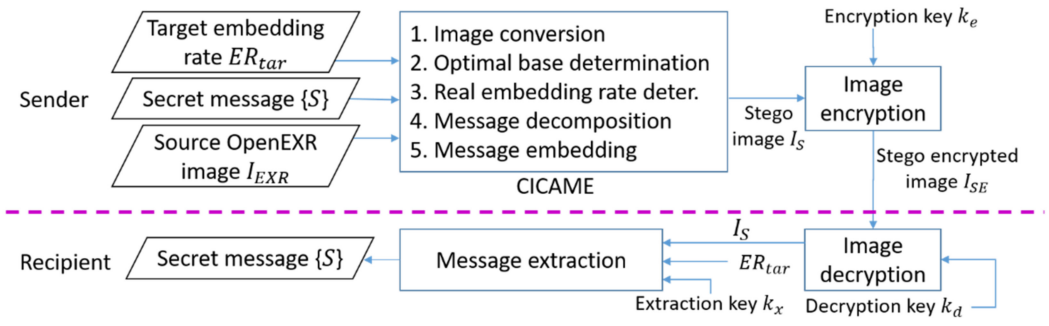


Figure 1. The flowchart of our proposed method consisting of message embedding and extraction processes.

To embed a secret message, the sender inputs the demanded target embedding rate (ER_{tar}), secret message $\{S\}$, and a source HDR OpenEXR image (I_{EXR}). The first part of message embedding is converting I_{EXR} into a new HDR image encoded by the RGBE format. During the conversion, we seamlessly embed secret message $\{S\}$ into each pixel to generate the stego image (I_S). Next, we encrypt I_S using the encryption key (k_e) to protect the image contents and secret hidden message. The resultant stego encrypted image (I_{SE}) is then delivered to the recipient through a public channel, such as the Internet. In this way, the secret message is camouflaged during the format conversion and image encryption.

Image decryption and message extraction are conducted on the recipient side. The receiver first inputs the stego encrypted HDR RGBE image (I_{SE}). It is then deciphered by the aid of the decryption key (k_d) provided prior by the sender. The image decryption produces the stego HDR RGBE image (I_S) which contains a hidden message. Finally, a secret message $\{S\}$ is extracted from I_S using the extraction key (k_x), thereby completing the message extraction process. The following subsections detail each step in the message embedding and extraction processes.

2.1. Concurrent Image Conversion and Adaptive Message Embedding

In this process, we convert the input HDR image, I_{EXR}, encoded by the OpenEXR format into the HDR image encoded by the RGBE format. Figure 2 shows the 48-bit/pixel OpenEXR and 32-bit/pixel RGBE formats. A pixel in an OpenEXR image is represented by a floating-point vector, $X = (\boldsymbol{X}_R,\ X_G,\ X_B)$. The floating-point value in the corresponding Red, Green, and Blue channels can be converted from 1-bit sign, 5-bit exponent, and 10-bit mantissa fields. In contrast, a pixel in an RGBE image is denoted by $\boldsymbol{P} = (P_R,\ P_G,\ P_B,\ P_E)$, representing the respective Red, Green, Blue, and the Exponent channels with 8-bit storage. During the conversion, we concurrently and adaptively embed a secret message, thus constructing a new stego HDR RGBE image, I_S. The process is completed by the following five steps:

- **Step 1:** Image conversion. An I_{EXR} is converted into the HDR image encoded by the RGBE format. The expressions shown on the left expression in Equation (1) convert chromatic information into the four components in the RGBE format:

$$\begin{cases} P_E = \lceil \log_2[\max(X_R, X_G, X_B)] + 128 \rceil, \\ P_R = \left\lfloor \dfrac{256 \times X_R}{2^{P_E - 128}} \right\rfloor, \\ P_G = \left\lfloor \dfrac{256 \times X_G}{2^{P_E - 128}} \right\rfloor, \\ P_B = \left\lfloor \dfrac{256 \times X_B}{2^{P_E - 128}} \right\rfloor, \end{cases} \qquad \begin{aligned} X_R &= \frac{P_R + 0.5}{256} \times 2^{P_E - 128} \\ X_G &= \frac{P_G + 0.5}{256} \times 2^{P_E - 128}, \\ X_B &= \frac{P_B + 0.5}{256} \times 2^{P_E - 128}, \end{aligned} \tag{1}$$

  where $\mathbf{X} = (X_R,\ X_G,\ X_B)$ represents the i-th pixel in scene-referred color in I_{EXR} and $\boldsymbol{P} = (P_R,\ P_G,\ P_B,\ P_E)$ denotes the i-th pixel in the corresponding RGBE HDR image. Note that we omit the index i if the expression does not cause any ambiguity. In a special case when $\max(X_R,\ X_G,\ X_B)$ is less than 10^{-38}, the conversion is written out as $(0,\ 0,\ 0,\ 0)$. We can translate the integer representation in $\boldsymbol{P} = (P_R,\ P_G,\ P_B,\ P_E)$ back to the scene-referred color, $\boldsymbol{X} = (X_R,\ X_G,\ X_B)$, using the right expressions in Equation (1), which is also known as reverse conversion. In a special case described above, we directly translate to $(0,\ 0,\ 0)$.

Note that the four components, $P_R,\ P_G,\ P_B,\ P_E$, are integers within the range of [0, 255]. While the first three components, $P_R,\ P_G,\ P_B$, represent the respective pixel values in the Red, Green, and Blue channels, the fourth component, P_E, indicates the pixel value in the Exponent channel (hereafter E-channel).

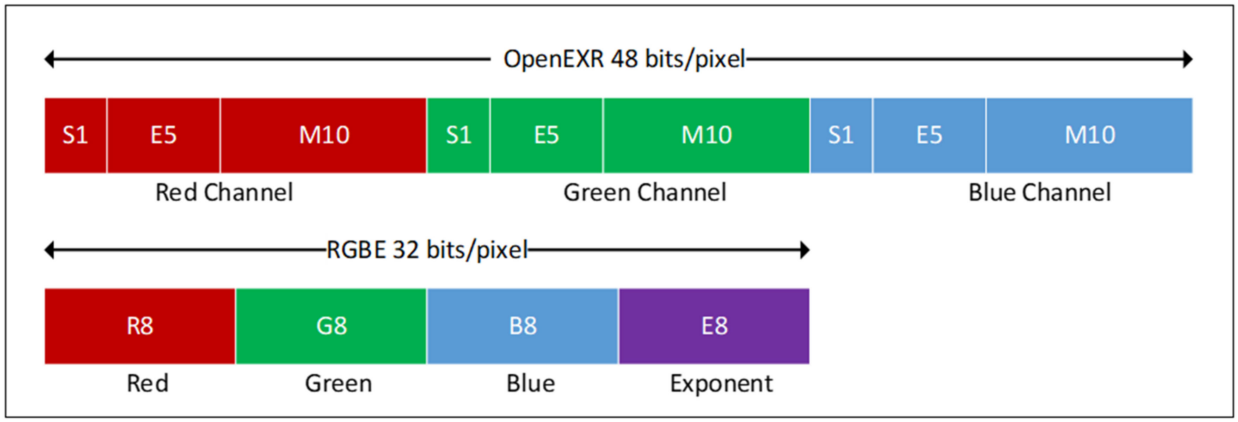


Figure 2. The bit breakdown for the 48-bit/pixel OpenEXR, 1-bit sign, 5-bit exponent, 10-bit mantissa, and 32-bit/pixel RGBE encodings, 8-bit in the Red, Green, Blue, and Exponent channel.

- **Step 2:** Optimal base determination. In this step, we determine an optimal base, $\boldsymbol{B} = (b_R, b_G, b_B)$, for the i-th pixel $\boldsymbol{P}$, according to P_E, i.e., this pixel's value in the E-channel. An optimal base is required so that we can embed a secret message in the P_R, P_G, P_B. In this paper, we convey a secret message in the Red-, Green-, and Blue-channels and leave the E-channel intact because the E-channel is the exponent shared by the other three channels, so any changes in it, for example, plus or minus n values $(\pm n)$, will result in the magnitude of exponential changes, $(\mathrm{R}, \mathrm{G}, \mathrm{B}) \times 2^{\pm n}$ in three other components. Instead, we utilize the value of the E-channel in a pixel $\boldsymbol{P}$ as a guide to determine its corresponding optimal base, thereby achieving adaptive message embedding. We detail this step as follows.

Let P_E $(0 \leq P_E \leq 255)$ represent the value of the pixel $\boldsymbol{P}$ in E-channel which is currently processed. Let E_m denote the median of the pixels in the E-channel derived from the HDR RGBE image which has just been converted from an OpenEXR HDR image. Let F represent the minimal notational number system where an F-ary secret message can be concealed in P_R, P_G, P_B. Inspired by the fact that human eyes are less sensitive to the pixel with low luminance (dark pixels), we derive F in Equation (2) as a function of P_E because the constant, ER_{tar}, is given by the end user and the constant, E_m, is derived from the HDR RGBE image,

$$F = \lceil 2^{ER_{tar}} \times 2^{E_m - P_E} \rceil. \tag{2}$$

The derived F strongly relates to the difference between P_E and the median E_m. If $P_E = E_m$, $F = 2^{ER_{tar}}$. This means that, if a pixel's P_E is identical to the median, the derived F equals the two's power of the target embedding rate $\left(2^{ER_{tar}}\right)$, as expected. However, if $P_E > E_m$, the derived F will be smaller than $2^{ER_{tar}}$, and the larger the P_E, the larger the difference between F and $2^{ER_{tar}}$; this means that fewer secret messages are conveyed in this pixel. In contrast, if $P_E < E_m$, the derived F will be larger than $2^{ER_{tar}}$, and the smaller the P_E, the larger the difference between F and $2^{ER_{tar}}$, implying that more secret messages will be conveyed in this pixel. This approach coincides with the concept that since human eyes are less sensitive to the pixel with low luminance (dark pixels), more secret message can be concealed in a pixel with smaller P_E (larger difference between E_m and P_E) without causing noticeable difference. In this paper, we set $ER_{tar} = 7.3$ bpp, and apparently, the end user can alter it without any restraint.

Once F is determined, we can derive an optimal base, $\boldsymbol{B} = (b_R, b_G, b_B)$ provided that two conditions are held. First, $b_R \times b_G \times b_B \geq F$; Second, (b_R, b_G, b_B) has minimal *expected* mean squared error (*EMSE*), which can be determined by Equation (3):

$$EMSE(\boldsymbol{B}) = \frac{1}{3 \times 12} \left\{ \left[(b_R)^2 - (-2)^{(b_R+1) mod 2} \right] + \left[(b_G)^2 - (-2)^{(b_G+1) mod 2} \right] + \left[(b_B)^2 - (-2)^{(b_B+1) mod 2} \right] \right\}. \tag{3}$$

We adopt the term, *expected*, implying that we assume the three components, b_R, b_G, b_B, have an equal probability of appearing. We remark that EMSE represents the distortion in the R-, G-, B-channels rather than the entire HDR RGBE image. We also remark that, since $\boldsymbol{B}$ depends on F, $\boldsymbol{B}$ is also a function of P_E. Once $\boldsymbol{B} = (b_R,\ b_G,\ b_B)$ is discovered, we can compute $M_{P_E} = b_R \times b_G \times b_B$, which may be equal to or larger than F. Consequently, M_{P_E} represents the ultimate notational system that a secret message can be concealed in $P_R,\ P_G,\ P_B$.

We present an example to further illustrate how to determine an optimal base. If an end user demands a target embedding rate, $ER_{tar} = 7.3$ bpp, and the median of the HDR RGBE image is $E_m = 126$. We consider four representative pixels, ***P*1**, ***P*2**, ***P*3**, ***P*4**, as follows: ***P*1** $= (165,\ 85,\ 16,\ 126)$, ***P*2** $= (195,\ 89,\ 37,\ 126)$, ***P*3** $= (85,\ 128, 48,\ 125)$ and ***P*4** $= (195,\ 89,\ 37,\ 129)$. First, since $P_E = 126$ in ***P*1** and this value equals the median, the corresponding $F = \lceil 2^{7.3} \times 2^{126-126} \rceil = 158$. Thus, an optimal base is $\boldsymbol{B} = (b_R,\ b_G,\ b_B) = (5,\ 5,\ 7)$ because $5 \times 5 \times 7 = 175 > F$ and $EMSE(\boldsymbol{B}) = \frac{1}{3\times 12}[24 + 24 + 48] = 2.667$ is minimal. Since $M_{P_E} = b_R \times b_G \times b_B = 175$, ***P*1** can conceal a 175-ary secret message. We remark that although other bases, such as $(5,\ 6,\ 6)$ or $(5,\ 6,\ 7)$, do satisfy the first condition, $5 \times 6 \times 6 \geq F$ and $5 \times 6 \times 7 \geq F$, unfortunately, they hold larger expected mean squared errors, where $EMSE(\boldsymbol{B1}) = 2.778$ and $EMSE(\boldsymbol{B2}) = 3.056$. Consequently, $\boldsymbol{B} = (5,\ 5,\ 7)$ does actually represent an optimal base for ***P*1**. Since ***P*2** and ***P*1** have the same P_E, they share the same optimal base able to conceal a 175-ary secret message.

With regard to the ***P*3** pixel, we derive $F = \lceil 2^{7.3} \times 2^{126-125} \rceil = 316$ and the optimal base is $\boldsymbol{B} = (7,\ 7,\ 7)$, thereby producing $EMSE(\boldsymbol{B})$ =4.0. Since $M_{P_E} = 7 \times 7 \times 7 = 343$, the ***P*3** pixel can carry a 343-ary secret message, conforming to our design goal for embedding a more secret message when P_E is smaller than E_m. Finally, an optimal base for the ***P*4** pixel is $\boldsymbol{B} = (3,\ 3,\ 3)$ because $F = \lceil 2^{7.3} \times 2^{126-129} \rceil = 20$ and $M_{P_E} = 27$. As expected, a limited 27-ary secret message will be concealed in this pixel, as its $P_E = 129$ is much larger than the median, $E_m = 126$.

- **Step 3:** Real embedding rate determination. Deriving the real embedding rate, ER_{rea}, to comply with the end user's request, we verify that the real embedding rate is greater than or equals the target embedding rate ($ER_{rea} \geq ER_{tar}$). In this step, we derive ER_{rea} using Equation (4):

$$ER_{rea} = \frac{1}{H \times V} \sum_{P_E=0}^{255} N_{P_E} \times log_2(M_{P_E}), \tag{4}$$

  where N_{P_E} represents the number of pixels in the entire HDR RGBE image holding P_E value in E-channel; for example, N_{132} denotes the number of pixels holding $P_E = 132$. Since P_E is within the range of [0, 255], it is certain that $N_{P_E} = N_0 + N_1 \ldots + N_{255}$ equals the image resolution of $H \times V$. In case $ER_{rea} < ER_{tar}$, we adjust the median E_m until we adhere to the end user's request. Finally, we will construct an E-channel mapping table, as shown in Table 1a, which maps every P_E value to N_{P_E}, F, M_{P_E}, $\boldsymbol{B} = (b_R,\ b_G,\ b_B)$, and $EMSE(\boldsymbol{B})$. In addition, we can derive the real embedding rate, ER_{rea}, and the expected mean squared error for three channels, $EMSE(RGB)$.

Table 1a presents the E-channel mapping table constructed after Step 2 for the HDR image "memorial". The end user requests the embedding rate of $ER_{tar} = 7.3$ bpp and the median of E-channel in this image is $E_m = 126$. We list N_{P_E} according to P_E in ascending order, and the associate optimal bases. Referring to Equation (4), the real embedding rate, $ER_{Rea} = 7.578$ bpp, is larger than the user's request. Also shown in the right part of table is the E-channel mapping table built when the requested embedding rate is $ER_{tar} = 9.0$ bpp. As can be seen from the table, when a user requests different ER_{tar}, the same P_E corresponds to different N_{P_E}, F, M_{P_E}, $\boldsymbol{B} = (b_R,\ b_G,\ b_B)$, and $EMSE(\boldsymbol{B})$. Nevertheless, both cases show that our scheme offers a more embedding rate than that request by the end user.

Table 1. (**a**) An example of the optimal base determination derived in the E-channel mapping table for the "memorial" mage with the median, $P_E = 126$. (**b**) An example of the optimal base determination derived in the E-channel mapping table for the "display1000" image with the median, $P_E = 125$.

(a)

P_E	N_{P_E}	F	M_{P_E}	b_R	b_G	b_B	$EMSE(B)$	F	M_{P_E}	b_R	b_G	b_B	$EMSE(B)$
≤119	0	-	-	-	-	-	-	-	-	-	-	-	-
120	1	10,086	10,143	21	21	23	39.111	32,768	32,768	32	32	32	85.500
121	1380	5043	5202	17	17	18	25.056	16,384	16,875	25	25	27	54.889
122	15,766	2523	2535	13	13	15	15.556	8192	8400	20	20	21	34.556
123	24,177	1261	1331	11	11	11	10.000	4096	4096	16	16	16	21.500
124	38,229	631	576	8	9	9	6.278	2048	2197	13	13	13	14.000
125	77,910	316	343	7	7	7	4.000	1024	1100	10	10	11	9.000
126	**108,287**	**158**	**175**	**5**	**5**	**7**	**2.667**	**512**	**512**	**8**	**8**	**8**	**5.500**
127	79,791	79	80	4	4	5	1.667	256	294	6	7	7	3.722
128	26,106	40	45	3	3	5	1.111	128	150	5	5	6	2.389
129	9609	20	27	3	3	3	0.667	64	64	4	4	4	1.500
130	3231	10	12	2	2	3	0.556	32	36	3	3	4	0.944
131	1819	5	6	1	2	3	0.389	16	18	2	3	3	0.611
132	1325	3	3	1	1	3	0.222	8	12	2	2	3	0.556
133	1711	2	2	1	1	2	0.167	4	4	1	2	2	0.333
134	2862	1	1	1	1	1	0	2	2	1	1	2	0.167
135	517	1	1	1	1	1	0	1	2	1	1	2	0.167
136	296	1	1	1	1	1	0	1	1	1	1	1	0
137	165	1	1	1	1	1	0	1	1	1	1	1	0
138	34	1	1	1	1	1	0	1	1	1	1	1	0
≥139	0	-		-	-	-	-	-		-	-	-	-
		ER_{tar}= 7.3, $ER_{rea} = 7.578$, $EMSE(RGB) = 3.900$						$ER_{tar} = 9.0$, $ER_{rea} = 9.251$, $EMSE(RGB) = 8.525$					

(b)

P_E	N_{P_E}	F	M_{P_E}	b_R	b_G	b_B	$EMSE(B)$	F	M_{P_E}	b_R	b_G	b_B	$EMSE(B)$
≤106	0	-	-	-	-	-	-	-		-	-	-	-
107	3	41,310,352	16,777,216	256	256	256	5461.500	134,217,728	16,777,216	256	256	256	5461.500
108~116	0	-	-	-	-	-	-	-	-	-	-	-	-
117	4	40,343	40,460	34	34	35	98.333	131,072	132,651	51	51	51	216.667
118	87	20,172	20,412	27	27	28	62.278	65,536	65,600	40	40	41	135.667
119	1755	10,086	10,143	21	21	23	39.111	32,768	32,768	32	32	32	85.500
120	6679	5043	5202	17	17	18	25.056	16,384	16,875	25	25	27	54.889
121	45,803	2522	2535	13	13	15	15.556	8192	8400	20	20	21	34.556
122	274,663	1261	1331	11	11	11	10.000	4096	4096	16	16	16	21.500
123	600,771	631	648	8	9	9	6.278	2048	2197	13	13	13	14.000
124	584,862	316	343	7	7	7	4.000	1024	1100	10	10	11	9.000
125	**649,392**	**158**	**175**	**5**	**5**	**7**	**2.667**	**512**	**512**	**8**	**8**	**8**	**5.500**
126	461,460	79	80	4	4	5	1.667	256	294	6	7	7	3.722
127	199,209	40	45	3	3	5	1.111	128	150	5	5	6	2.389
128	123,106	20	27	3	3	3	0.667	64	64	4	4	4	1.500
129	83,390	10	12	2	2	3	0.556	32	36	3	3	4	0.944
130	51,408	5	6	1	2	3	0.389	16	18	2	3	3	0.611
131	41,206	3	3	1	1	3	0.222	8	12	2	2	3	0.556
132	17,509	2	2	1	1	2	0.167	4	4	1	2	2	0.333
133	4421	1	1	1	1	1	0	2	2	1	1	2	0.167
≥134	0	-		-	-	-	-	-		-	-	-	-
		ER_{tar}= 7.3, $ER_{rea} = 7.618$, $EMSE(RGB) = 4.041$						$ER_{tar} = 9.0$, $ER_{rea} = 9.293$, $EMSE(RGB) = 8.837$					

Table 1b shows another E-channel mapping table constructed for the HDR image "display1000". This image has the median of E-channel, $E_m = 125$. We can observe from the table that, when a pixel has a smaller P_E, a larger optimal base is determined, thereby enabling it to conceal a larger M_{P_E}-ary secret message. Again, the real embedding rate, $ER_{rea} = 7.618$ bpp, is larger than the request, $ER_{tar} = 7.3$ bpp.

These two tables demonstrate that our scheme can adaptively conceal a secret message according to the features of pixels in an HDR RGBE image. In addition, the overall real embedding rate is always larger than the one requested by the end user, yet our scheme ensures determining an optimal base, thereby producing a stego image with the minimal expected mean squared error.

- **Step 4:** Message decomposition. Given a general n-tuple optimal base $\boldsymbol{B} = (b_1, b_2, \ldots, b_n)$ and an M-ary secret message, S_M, where $M = \prod_{i=1}^{n} b_i$, we can decompose S_M into n secret digits $(d_1, d_2, \ldots, d_n)$ using Equation (5):

$$d_i = \begin{cases} S_M \bmod b_i, & if\ i = 1 \\ \left\lfloor \frac{S_M}{\prod_{j=1}^{i-1} b_j} \right\rfloor \bmod b_i, & if\ 2 \le i \le n \end{cases} \tag{5}$$

In this paper, we produce a 3-tuple optimal base $\boldsymbol{B} = (b_R, b_G, b_B)$, so we set $n = 3$ in Equation (5) to decompose an M_{P_E}-ary secret message, $S_{M_{P_E}}$, into three digits, (d_1, d_2, d_3). As an example, given $\boldsymbol{B} = (5, 5, 7)$ and a 175-ary secret message, $S_{175} = 89$, we can decompose 89 into 3 digits as follows. First, we let $i = 1$ and derive $d_1 = 89 \bmod 5 = 4$. Next, we derive $d_2 = \left\lfloor \frac{89}{5} \right\rfloor \bmod 5 = 2$. Finally, $i = 3$, we obtain $d_3 = \left\lfloor \frac{89}{5\times 5} \right\rfloor \bmod 7 = 3$. Consequently, the three digits decomposed by $\boldsymbol{B} = (5, 5, 7)$ are $(d_1, d_2, d_3) = (4, 2, 3)$.

- **Step 5:** Message embedding. Once we have obtained (d_1, d_2, d_3), we can embed them into (P_R, P_G, P_B) by the aid of the optimal base (b_R, b_G, b_B), thus producing a 3-tuple stego pixel components,(P'_R, P'_G, P'_B). The concept of embedding (d_1, d_2, d_3) is producing (P'_R, P'_G, P'_B) so that three digits can later be extracted using the modulus operator. In addition, we need to minimize the distortion, $\left|(P'_R, P'_G, P'_B) - (P_R, P_G, P_B)\right|$, caused by the message embedding. We take the component P_R and the digit d_1 as an example to illustrate the digit embedding, which utilizes Equations (6)–(8) to produce the stego component P'_R:

$$r_R = P_R \bmod b_R \tag{6}$$

$$v_R = [(d_1 - r_R) + b_R] \bmod b_R \tag{7}$$

$$P'_R = \begin{cases} P_R, & if\ v_R = 0 \\ P_R + v_R, & if\ 0 < v_R < \left\lfloor \frac{b_R}{2} \right\rfloor \\ P_R + v_R - b_R, & if\ \left\lfloor \frac{b_R}{2} \right\rfloor \le v_R < b_R \end{cases} \tag{8}$$

In Equation (6), we first obtain the remainder r_R using the divisor b_R, and then calculate the difference, v_R, between d_1 and r_R in Equation (7). Finally, referring to Equation (8), we produce P'_R according to the magnitude of v_R with respect to b_R. Note that the component P_E in the E-channel remains intact during the digit embedding. We can follow the same approach to embed other two digits, d_2 and d_3, accordingly.

We now analyze the time complexity of the embedding process. Let $H \times V$ represent the size of an HDR RGBE image. We remark that the message concealment consists of five steps, as described above, and it is performed in the Red, Green, and Blue channels in a pixel-by-pixel approach. Consequently, the time complexity of message concealment is $O(H \times V \times 3)$. We now present an example for message embedding.

We present an example to illustrate the message embedding process. Let $\boldsymbol{X} = (0.161621, 0.083496, 0.016113)$ represent a pixel in the OpenEXR image. In step 1, we convert $\boldsymbol{X}$ to be a pixel $\boldsymbol{P}$ in the RGBE format, where $P = (165, 85, 16, 126)$. In step 2, without loss of generality, we assume the optimal base has been determined, where $\boldsymbol{B} = (b_R, b_G, b_B) = (5, 5, 7)$. In step 3, we determine the real embedding rate as shown in Table 1a. In step 4, the message, $S_{175} = 89$, has been decomposed into $(d_1, d_2, d_3) = (4, 2, 3)$. To embed these three digits in step 5, we first refer to Equation (6) and obtain the remainders at each component, where $(r_R, r_G, r_B) = (0, 0, 2)$. We then calculate the difference, $(v_R, v_G, v_B) = (4, 2, 1)$, in Equation (7). Finally, using Equation (8), we derive the stego components in the Red, Green, and Blue channels, $(P'_R, P'_G, P'_B) = (164, 87, 17)$. Thus, the stego pixel produced becomes $\boldsymbol{P'} = (P'_R, P'_G, P'_B, P'_E) = (164, 87, 17, 126)$, as the E-channel is intact. All the other pixels can work in the same manner to convey the corresponding digits.

2.2. Image Encryption

This stego image encryption process ciphers the stego image to protect secret messages and avoid the image from unauthorized user access. The image encryption contains the following four steps:

- **Step 1**: Obtaining an image feature h. We calculate the feature of a stego HDR RGBE image (h) in Equation (9):

$$h = \frac{\sum_{i=1}^{4\times H\times V} b_i}{255 \times 4 \times H \times V} \tag{9}$$

  where b_i represents the i-th byte data of the image. The parameter h is considered as a part of the secret encryption key, where its range is within [0, 1].
- **Step 2:** Generating a pseudo-random sequence. In this step, we produce a pseudo-random sequence from the 2D-SLMM [33]. As shown in Equation (10):

$$\begin{cases} x_{i+1} = \alpha(\sin(\pi y_i) + \beta)x_i(1 - x_i) \\ y_{i+1} = \alpha(\sin(\pi x_{i+1}) + \beta)y_i(1 - y_i) \end{cases} \tag{10}$$

The range of the control parameters, α and β, is $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 3$, respectively. When the parameter β is close to 3, 2D-SLMM shows good hyperchaotic performance. In this study, we select $(\alpha,\ \beta) = (1,\ 3)$. The trajectories of the 2D-SLMM map are shown in Figure 3, which provides a wider chaotic range, better ergodicity, and hyperchaotic properties than existing chaotic maps. In addition, we set the initial value $(x_0,\ y_0) = (x_K \times h,\ y_K \times h)$, where (x_K, y_K) represents a part of the secret encryption key. Note that the initial value is related to the feature of an image derived from Step 1, indicating that the initial values dynamically depend on the stego HDR RGBE image to be processed, thereby increasing the security of image encryption.

Given α, β, and $(x_0,\ y_0)$, we generate $q + 4 \times H \times V$ number of pseudo-random sequence, $\boldsymbol{R} = \{R_0, R_1, \ldots, R_q, R_{q+1}, \ldots, R_{q+4\times H\times V-1}\}$, where q is the number of elements in the sequences to be discarded in order to eliminate the transient effects. Note that each element R_i is produced by first computing $x_i \times y_i$ and then representing the values as the IEEE double-precision floating-point format (IEEE Standard for Floating-Point Arithmetic, IEEE 754) [34], followed by extracting the least significant 8 bits to form an integer between 0 and 255.

- **Step 3:** Pixel bit-level permutation and diffusion. We adopt a secret encryption key, k_e, to perform bit-level permutation, which shuffles both the pixel contents (32 bits) and the pixel positions in the stego HDR RGBE image. We adopt a random permutation scheme [35] with an encryption key to generate pseudo-random numbers to accomplish the bit-level permutation.

We take an 8-bit pixel as an example. Let $\{\pi\} = \{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8\}$ represent the index of position in a pixel, and $\{D\} = \{0,\ 1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 1\}$ denote the corresponding bits, which represent the decimal value 91. Thus, the first three bits, for example, can be referred to by $\pi[1] = 0$, $\pi[2] = 1$, and $\pi[3] = 0$, etc. When utilizing the random permutation scheme, we execute 7 iterations. At the i-th iteration, we generate a $(9 - i)$-ary random number k. Then, we exchange the current last index with the index k. The process continues 7 times until we shuffle the indices in $\{\pi\}$ completely.

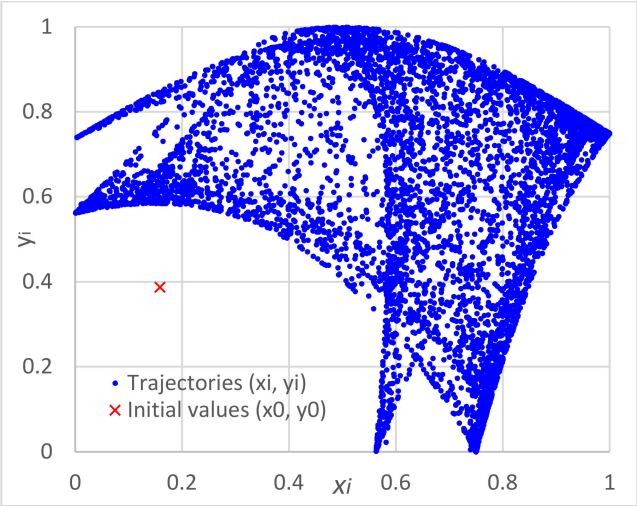


Figure 3. Trajectories of 2D-SLMM with an initial value $(x_0,\ y_0) = (0.15874,\ 0.38797)$ and the control values $(\alpha,\ \beta) = (1,\ 3)$.

Table 2 shows an example of the index permutation, where the shaded color indicates indices that have been shuffled. In the first round (i = 1), we generate an 8-ary random number $k = 3_8$, and then exchange the 3rd index, 3, and the last index, which is 8, thus producing $\{\pi\} = \{1,\ 2,\ 8,\ 4,\ 5,\ 6,\ 7,\ \mathbf{3}\}$. In the second iteration (i = 2), we generate a 7-ary random number, say $k = 4_7$; we exchange the 4th index, 4, and the last index, which is 7, thus leading to $\{\pi\} = \{1,\ 2,\ 8,\ 7,\ 5,\ 6,\ \mathbf{4},\ \mathbf{3}\}$. In the third iteration (i = 3), we generate a 6-ary random number from 1 to 5, say $k = 1_6$; we exchange the first index 1 and the last index, which is 5, thus leading to $\{\pi\} = \{6,\ 2,\ 8,\ 7,\ 5,\ \mathbf{1},\ \mathbf{4},\ \mathbf{3}\}$. The final permuted index $\{\pi\} = \{5,\ 8,\ 2,\ 6,\ 7,\ 1,\ 4,\ 3\}$, implying that the bit-level permutation produces $\{D\} = \{1,\ 1,\ 1,\ 0,\ 1,\ 0,\ 1,\ 0\}$ representing a decimal value 234.

Table 2. An example of random index permutation, where $\{\pi\} = \{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8\}$ represents the original index order corresponding to an 8-bit pixel.

i	k	Exchange	1st	2nd	3rd	4th	5th	6th	7th	8th
Original index $\{\pi\}$			1	2	3	4	5	6	7	8
1	3_8	(3, 8)	1	2	8	4	5	6	7	3
2	4_7	(4, 7)	1	2	8	7	5	6	4	3
3	1_6	(1, 6)	6	2	8	7	5	1	4	3
4	4_5	(5, 7)	6	2	8	5	7	1	4	3
5	1_4	(5, 6)	5	2	8	6	7	1	4	3
6	2_3	(2, 8)	5	8	2	6	7	1	4	3
7	0_2	No change	5	8	2	6	7	1	4	3
Permuted index $\{\pi\prime\}$			5	8	2	6	7	1	4	3

In our implementation, we align a total of $4 \times H \times V$ components (each pixel contains the R-, G-, B- and E-channel components). We then apply the bit-level random permutation, described above, to all of the bits. We remark that the random permutation shuffles the indices in place rather than producing a shuffled copy. Along with the increase of iterations, the number of indices in the shuffled set increases. Consequently, the random permutation algorithm has the time complexity of $O(N)$ and the space complexity of $O(1)$.

- **Step 4:** Pixel diffusion. The final step is to encrypt the shuffled pixels using exclusive-OR operator, $\oplus$, with respect to the pseudo-random sequence, $\boldsymbol{R\prime} = \{R_q, R_{q+1}, \ldots, R_{q+4\times H\times V-1}\}$. Without loss of generality, we list the shuffled pixel as $I'_S = \{P_{1,R'},\ P_{1,G'},\ P_{1,\ B'},\ P_{1,E'}, \ldots,\ P_{H\times V,R'}, P_{H\times V,G'}, P_{H\times V,B'}, P_{H\times V,E'}\}$. The pixel encryption contains two sub-steps. First, we generate five initial values, $\{P_{0,R'},\ P_{0,G'},\ P_{0,\ B'},\ P_{0,E'},\ C_{0,E'}\}$, from the encryption

key. Next, we utilize Equation (11) to cipher four components. Note that since we have provided five initial values, the index ranges from 0 to $H \times V$.

$$\begin{cases} C_{i+1,\,R'} = R'_{4i} \oplus P_{i+1,R'} \oplus P_{i,R'} \oplus C_{i,E'} \\ C_{i+1,G'} = R'_{4i+1} \oplus P_{i+1,G'} \oplus P_{i,G'} \oplus C_{i+1,R'} \\ C_{i+1,B'} = R'_{4i+2} \oplus P_{i+1,B'} \oplus P_{i,B'} \oplus C_{i+1,G'} \\ C_{i+1,E'} = R'_{4i+3} \oplus P_{i+1,E'} \oplus P_{i,E'} \oplus C_{i+1,B'} \end{cases} \tag{11}$$

As an example, let $\boldsymbol{P}'_1 = \left(P_{1,R'},\, P_{1,G'},\, P_{1,\,B'},\, P_{1,E'}\right) = (164,\, 87,\, 17,\, 126)$ represent the first stego pixel and pseudo-random sequence $\boldsymbol{R'} = \{38,\, 39,\, 246,\, 133,\, 151, \ldots\}$. Let $\left(P_{0,R'},\, P_{0,G'},\, P_{0,\,B'},\, P_{0,E'},\, C_{0,E'}\right) = (196,\, 118,\, 25,\, 57,\, 49)$ be the initial five values. Referring to Equation (11) with the index $i = 0$, we can derive $C_{1,\,R'} = 38 \oplus 164 \oplus 196 \oplus 49 = 119$. We can further adopt the resultant $C_{1,\,R'}$ to encrypt $C_{1,G'}$, thus producing $C_{1,G'} = 39 \oplus 87 \oplus 118 \oplus 119 = 113$. Similarly, we can encrypt $C_{1,B'} = 143$ using the previous result, $C_{1,G'}$; we can encrypt $C_{1,E'} = 77$ using the previous result, $C_{1,B'}$, accordingly. Consequently, the stego pixel, $\boldsymbol{P}'_1 = (164,\, 87,\, 17,\, 126)$, has been ciphered to become $\boldsymbol{C}'_{\boldsymbol{1}} = (119,\, 113,\, 143,\, 77)$.

Our scheme provides the benefit of the avalanche effect [23,24], which means that an error encountered in a previous component will produce an error in the current component being processed. This chain-reaction feature will be propagated across the entire components to be encrypted, causing a drastic change in the ciphertext stego HDR RGBE image.

We now analyze the time complexity of the image encryption process. In the image encryption, we first obtain an image feature h, which requires the time complexity of $O(H \times V \times 4)$. Next, referring to a 2D-SLMM map, we generate a pseudo-random sequence, which needs the time complexity of $O(q + H \times V \times 4)$, where q is used to avoid the transient effects. The third step performs the bit-level permutation in all four channels, thus requiring the time complexity of $O(4 \times H \times V)$. In the final step, the pixel diffusion needs the time complexity of $O(H \times V \times 4)$.

In summary, the time complexity of four steps determines the computational complexity of our image encryption algorithm Consequently, our stego HDR encryption scheme has the computational complexity of $O(q + H \times V \times 16)$. We consider this complexity has satisfactory performance because there is an extra exponent channel in an HDR RGBE image.

2.3. Image Decryption

The decryption process is the inverse operation of the encryption process. The input is the stego encrypted HDR RGBE image, I_{SE}, and the output is the deciphered stego HDR RGBE image, I_S, which contains a hidden message.

The image decryption contains the following three steps:

- **Step 1:** We determine parameters, h, q, x_K, y_K from the secret decryption key. Then, we input them as initial values in the 2D-SLMM to generate the pseudo-random sequence $\boldsymbol{R} = \left\{R_0, R_1, \ldots, R_q,\, R_{q+1},\, \ldots,\, R_{q+4\times H\times V-1}\right\}$. We discard the first q items so the resultant $\boldsymbol{R'}$ is exactly the same as that produced in the encryption process.
- **Step 2:** We determine five setting values, $\left(P_{0,R'},\, P_{0,G'},\, P_{0,\,B'},\, P_{0,E'},\, C_{0,E'}\right)$ from the decryption key. Then, we utilize Equation (12) to decipher stego encrypted pixels in four components:

$$\begin{cases} P_{i+1,R'} = R'_{4i} \oplus C_{i+1,\,R'} \oplus P_{i,R'} \oplus C_{i,E'} \\ P_{i+1,G'} = R'_{4i+1} \oplus C_{i+1,G'} \oplus P_{i,G'} \oplus C_{i+1,R'} \\ P_{i+1,B'} = R'_{4i+2} \oplus C_{i+1,B'} \oplus P_{i,B'} \oplus C_{i+1,G'} \\ P_{i+1,E'} = R'_{4i+3} \oplus C_{i+1,E'} \oplus P_{i,E'} \oplus C_{i+1,B'} \end{cases} \tag{12}$$

Following up on our previous example, we present $\boldsymbol{R'} = \{38,\, 39,\, 246,\, 133,\, 151, \ldots\}$ and the setting values $\left(P_{0,R'},\, P_{0,G'},\, P_{0,\,B'},\, P_{0,E'},\, C_{0,E'}\right) = (196,\, 118,\, 25,\, 57,\, 49)$. Let $\boldsymbol{C}'_{\boldsymbol{1}} = (119,\, 113,\, 143,\, 77)$ be the first stego encrypted pixel. Referring to Equation (12), we produce $\boldsymbol{P}_{1,R'} = 38 \oplus 119 \oplus 196 \oplus 49 = 164$. We can apply the same manner to decipher the other three components. Thus, the decrypted stego pixel becomes $P'_1 = (164,\, 87,\, 17,\, 126)$.

- **Step 3:** Pixel bit-level inverse permutation and diffusion. The recipient can adopt the secret encryption key, k_e, to generate pseudo-random numbers to accomplish the inverse bit-level permutation. We follow up on the previous example. Let $\{\pi\prime\} = \{5, 8, 2, 6, 7, 1, 4, 3\}$ represent the index of position in an input pixel. When utilizing the inverse version of the permutation scheme, we also generate a $(9 - i)$-ary random number k for the i-th iteration. However, we apply these random numbers in a reverse order, i.e., let $j = 8 - i$ and j be the actual index we used in the inverse version. Then, we exchange index j+1 with index k. The process continues 7 times until the indices in $\{\pi\prime\}$ are fully permuted.

Table 3 continues the example for the inverse index permutation, where the shaded color indicates an index to be exchanged between itself and the index k, which is determined by the generated random number. In the first round ($j = 1$), the random number produced is $k = 0_2$, and no exchange is needed because it is in the 1st index. In the second iteration ($j = 2$), the random number produced is $k = 2_3$; we exchange the 2nd index, 8, and the $j + 1 = 3$rd index, which is 2, thus leading to the current $\{\pi\prime\} = \{5, 2, 8, 6, 7, 1, 4, 3\}$. In the third iteration ($j = 3$), the random number produced is $k = 1_4$; we exchange the 1st index, which is 5, and the 4th index, which is 6, thus updating $\{\pi\prime\}$ to become $\{\pi\prime\} = \{6, 2, 8, 5, 7, 1, 4, 3\}$. The final inverse permuted index $\{\pi\} = \{1, 2, 3, 4, 5, 6, 7, 8\}$, indicating that $\{D\} = \{0, 1, 0, 1, 1, 0, 1, 1\}$ represents the inverse bit-level permutation, which is the same as the original decimal pixel value 91.

Table 3. An example of inverse random index permutation, where $\{\pi\prime\} = \{5, 8, 2, 6, 7, 1, 4, 3\}$ represents the input index order corresponding to an 8-bit pixel.

j	k	Exchange	1st	2nd	3rd	4th	5th	6th	7th	8th
Input index $\{\pi\prime\}$			5	8	2	6	7	1	4	3
1	0_2	No change	5	8	2	6	7	1	4	3
2	2_3	(8, 2)	5	2	8	6	7	1	4	3
3	1_4	(5, 6)	6	2	8	5	7	1	4	3
4	4_5	(5, 7)	6	2	8	7	5	1	4	3
5	1_6	(6, 1)	1	2	8	7	5	6	4	3
6	4_7	(4, 7)	1	2	8	4	5	6	7	3
7	3_8	(3, 8)	1	2	3	4	5	6	7	8
Inverse index permutation results: $\{\pi\}$			1	2	3	4	5	6	7	8

2.4. Message Extraction

Since E-channel is intact and the target embedding rate ER_{tar} is considered as a secret key, the receiver can reconstruct the E-channel mapping table similar to Table 1. Then, every pixel, $\boldsymbol{P'} = (P'_R, P'_G, P'_B, P'_E)$, in the stego image can be processed pixel-by-pixel. For each pixel, the receiver refers to P'_E to obtain the optimal base (b_R, b_G, b_B) from the E-channel mapping table. Afterwards, one can extract the secret messages, (d_1, d_2, d_3), using the modulus operator with the divisor (b_R, b_G, b_B) for every component, where $(d_1, d_2, d_3) = (P'_R \ mod \ b_R, P'_G \ mod \ b_G, P'_B \ mod \ b_B)$. Finally, the receiver can apply Equation (13) with the parameter, $n = 3$, using the available (d_1, d_2, d_3) to composite the M-ary secret message S_M, where $M = \prod_{i=1}^{n} b_i$, and $\boldsymbol{B} = (b_1, b_2, \ldots, b_n)$ represents a general n-tuple optimal base:

$$S_M = d_1 + \sum_{i=2}^{n} \left[d_i \times \left(\prod_{j=1}^{i-1} b_j \right) \right] \tag{13}$$

Following up on our previous example, we first read in the stego pixel $\boldsymbol{P'_1} = (P'_R, P'_G, P'_B, P'_E) = (164, 87, 17, 126)$. Then, referring to the E-channel mapping table with $P_E = 126$, as shown in Table 1, we derive the optimal base, $(b_R, b_G, b_B) = (5, 5, 7)$. Next, we extract the message digits $(d_1, d_2, d_3) = (164 \ mod \ 5, 87 \ mod \ 5, 17 \ mod \ 7) = (4, 2, 3)$. Finally, applying Equation (13), we composite the final 175-ary secret message, $S_M = 4 + [2 \times (5) + 3 \times (5 \times 5)] = 89_{175}$.

3. Experimental Results and Analysis

We implemented our algorithm using C++ and Python programming languages and collected our experimental results on a Personal Computer equipped with an i7-10610U CPU, 16 GB RAM, and Windows 10 operating system. Unlike LDR data hiding, no standard image database is available for HDR image, and we instead collect our 20 test images form the Internet [36] and indexed them from test image 1 to test image 20. We remark that some of these HDR images have been adopted as test images in the literature [18–20,29,30]. In the following, we first report the secret message embedding results and then present the security analysis of the encrypted stego HDR images.

3.1. Secret Message Embedding Results

Figure 4 lists 20 stego HDR RGBE images produced by our scheme, each conveying a different number of secret bits. Note that these images are tone-mapped low dynamic range images, and we adopted the tone-mapping algorithms introduced in [2,37].

We present a larger size of the stego image "dani_catheral" for the purpose of visualization, as shown in Figure 5. This stego image has conveyed around 5.92 million of secret bits prior to the encryption process. As shown in Figure 5a, direct display of a stego HDR image is not satisfying because over- and under-exposed areas hide image features. In contrast, the tone-mapped (TM) stego images reveal their full details and give them a dynamic twist as well as a realistic look, as shown in Figure 5b,c, respectively, Nevertheless, even though our scheme offers a significant high embedding rate, reaching 7.550 bpp, the distortions in these stego images are difficult to be perceived by human eyes.

Table 4 shows the features of test images and the embedding capacity as well as the results of the image quality assessments. Depending on the image resolutions, the embedding capacity ranges from 51.25 million bits (No. 15, 3025×2129) to 2.59 million bits (No. 13, 720×480). Depending on the distribution of E-channel, the embedding rate ranges from 7.386 bpp (No. 11) to 8.258 bpp (No. 16). Nevertheless, all the test images comply with the demand of 7.3 bpp requested by the end user. As observed in Table 4, the mean squared error of three channels, MSE(RGB), in the stego HDR RGBE image varies from 3.183 to 45.546, depending on the image resolution, the distribution of the E-channel, and that of the secret message. In this table, we show the PSNR values of the tone-mapped images as they are now in the low dynamic range, thereby using 8 bits per sample to represent a pixel. Apart from the "Still Life" image, the PSNR values are all larger than 40 dB, while some of them are even higher than 50 dB. In addition, the structural similarity index measure (SSIM) and the universal image quality index (Q-index) are all very close to 1.0. Image assessment results indicate that the stego tone-mapped image has good image quality.

3.2. Security Analysis

This section evaluates the performance of image encryption when we encrypt the stego HDR RGBE image by our scheme to produce a stego encrypted HDR RGBE image, I_{SE}. To the best of our knowledge, no existing security metrics are available to evaluate an encrypted HDR image. Since the interest of our study focuses on the RGBE format with four channels, we instead modify the LDR security evaluation metrics extending them to four channels in an HDR RGBE image. In this study, we use six categories to analyze the security of the encrypted images: the visual perception, histogram, correlation, entropy, image sensitivity, and key security.

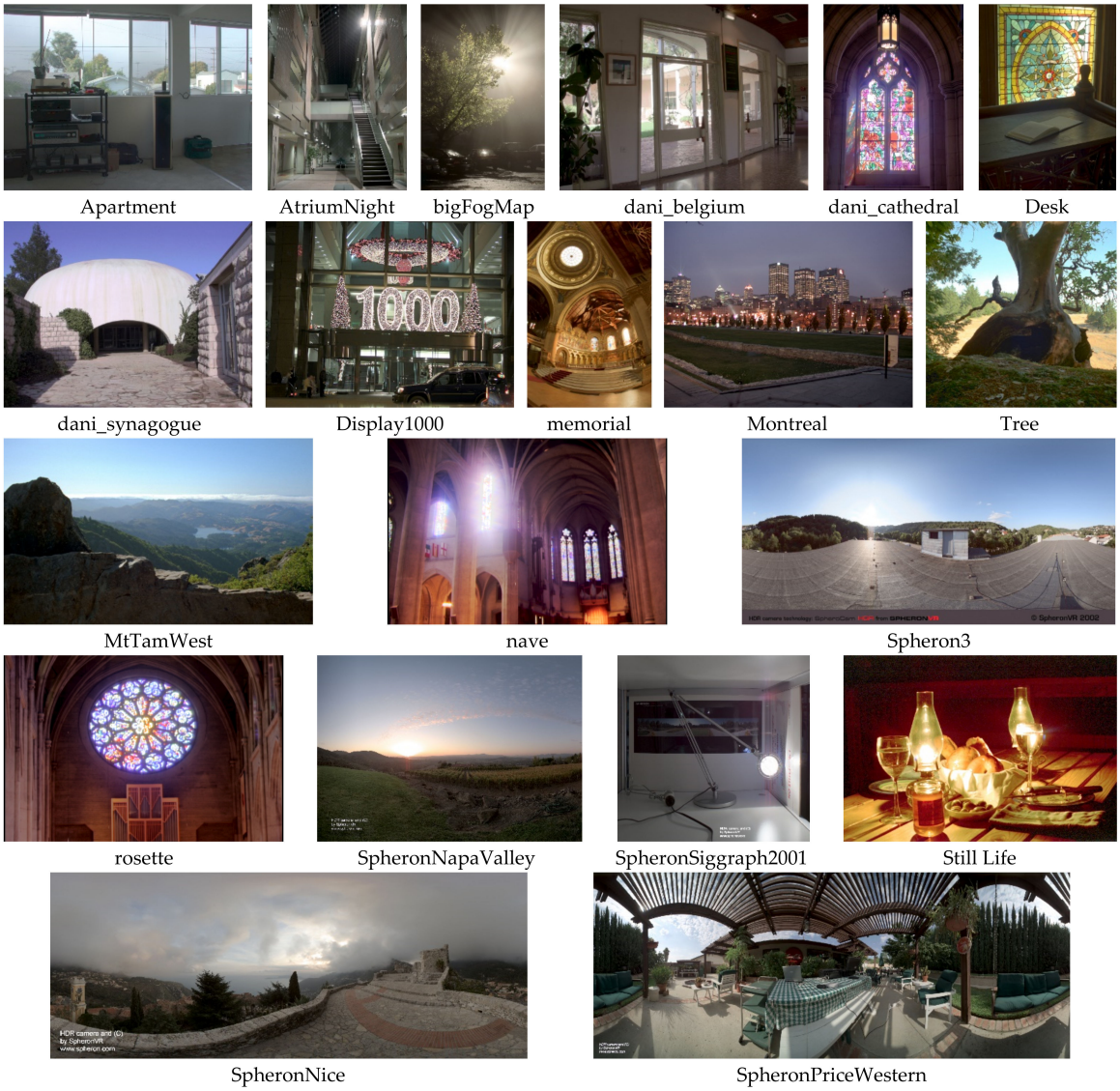
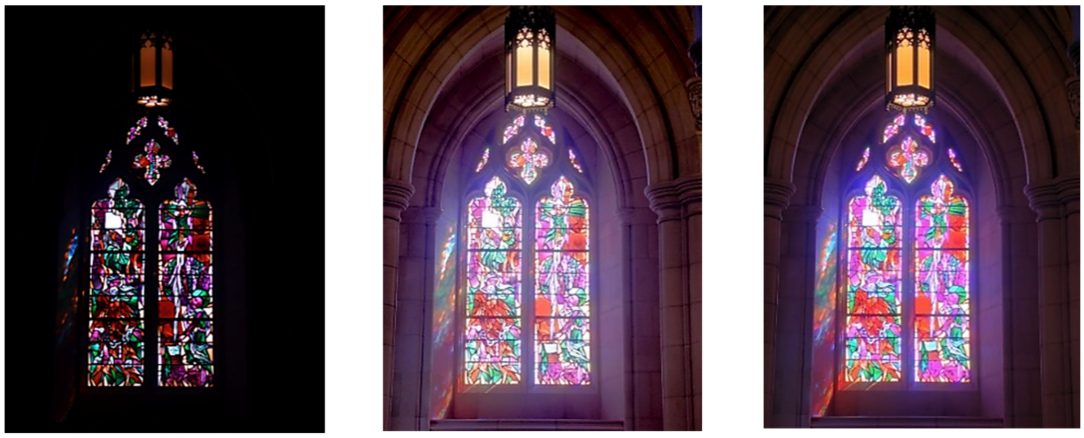


Figure 4. An exhibition of 20 stego tone-mapped images before the encryption process.



(**a**) Direct display (**b**) Stego TM image (**c**) Stego TM image

Figure 5. Stego HDR images "dani_cathedral": (**a**) direct display of an HDR image; (**b**) and (**c**) using different tone-mapped algorithms [37] and [2], respectively, and then displaying the stego low dynamic range TM images.

Table 4. Embedding capacity and quality assessment of the stego images before encryption.

ID	Name	Resolution	Capacity	ER_{rea} (bpp)	MSE (RGB)	TM PSNR	TM SSIM	TM Q-Index
1	Apartment	2048 × 1536	24,225,583	7.701	17.609	51.895	0.999918	0.996560
2	AtriumNight	760 × 1016	5,831,456	7.552	3.445	51.714	0.999786	0.999928
3	bigFogMap	751 × 1130	6,630,761	7.813	4.118	50.342	0.999629	0.997726
4	dani_belgium	1025 × 769	6,054,383	7.681	4.499	51.053	0.999689	0.998506
5	dani_cathedral	767 × 1023	5,924,398	7.550	4.723	49.988	0.999584	0.998657
6	dani_synagogue	1025 × 769	6,388,178	8.105	5.418	47.352	0.999364	0.999867
7	Desk	644 × 874	4,247,692	7.547	6.515	45.386	0.999687	0.990792
8	Display1000	2048 × 1536	23,962,622	7.618	4.041	50.545	0.999950	0.998025
9	memorial	512 × 768	2,979,918	7.578	3.900	42.611	0.999242	0.997763
10	Montreal	2048 × 1536	23,637,108	7.514	3.183	49.552	0.999869	0.999894
11	MtTamWest	1214 × 732	6,563,231	7.386	6.274	50.149	0.999601	1.000000
12	nave	720 × 480	2,639,315	7.637	42.353	46.835	0.998883	0.996546
13	rosette	720 × 480	2,597,575	7.516	45.546	47.052	0.998770	0.994346
14	Spheron3	2149 × 1074	18,556,270	8.040	16.753	40.906	0.998803	0.998281
15	SpheronNapaValley	3025 × 2129	51,250,510	7.958	5.321	43.923	0.999597	1.000000
16	SpheronNice	2981 × 1165	28,677,186	8.258	5.341	43.411	0.999470	0.999264
17	SpheronPriceWestern	3272 × 1280	32,678,515	7.803	4.570	51.209	0.999936	0.998134
18	SpheronSiggraph2001	1329 × 1289	12,676,596	7.400	3.634	49.223	0.999822	0.998210
19	StillLife	1240 × 846	8,002,655	7.629	25.401	32.602	0.989207	0.997025
20	Tree	928 × 906	6,922,330	8.233	35.164	50.494	0.999914	0.998838

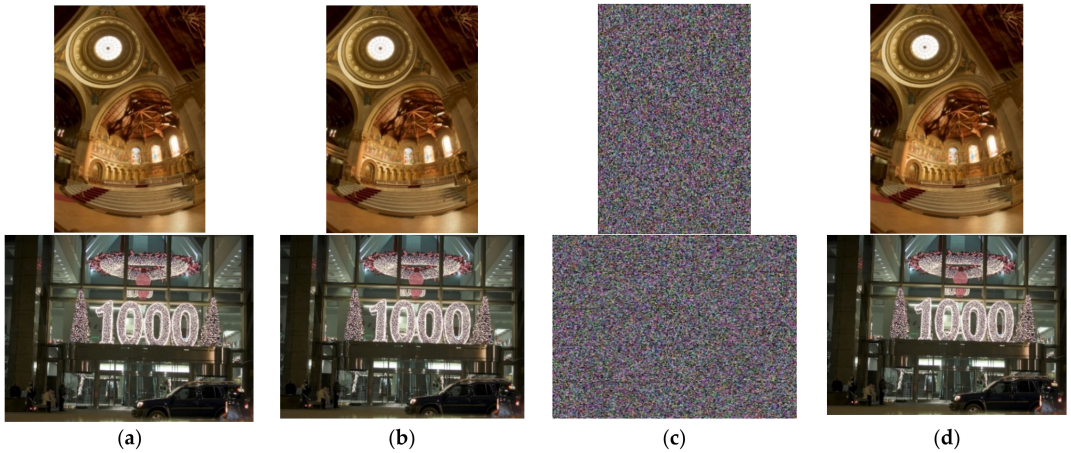
3.2.1. Visual Perception

Figure 6 exhibits the tone-mapped images generated from our scheme in different stages. We adopted the tone-mapping algorithm introduced in [2]. From Figure 6a,b, image distortion is not perceived after the format conversion and the message embedding. Figure 6c shows the encryption images which successfully shelter the outlines and detail features. The encrypted images are completely submerged by noise without revealing any meaningful information. Consequently, our encryption scheme is visually secure.

3.2.2. NIST SP 800-22 Randomness Test

The National Institute of Standards and Technology (NIST) statistical test [38] was adopted to test the randomness of sequences generated by the 2D-SLMM to ensure that they are suitable for cryptosystems. NIST SP 800-22 consists of 16 statistical tests, and each test provides a p-value between 0 and 1 under the significance level α. If p-value $\geq \alpha$, the sequence passes the randomness test successfully with the confidence of $1 - \alpha$. Otherwise, the sequence fails the test. In our experiment, we set the default value, $\alpha = 0.01$ for testing.

Table 5 shows the NIST test results. From the table, we can see that all p-values are larger than 0.01, indicating that the 2D-SLMM sequence passes the randomness tests with 99% confidence.



(**a**) (**b**) (**c**) (**d**)

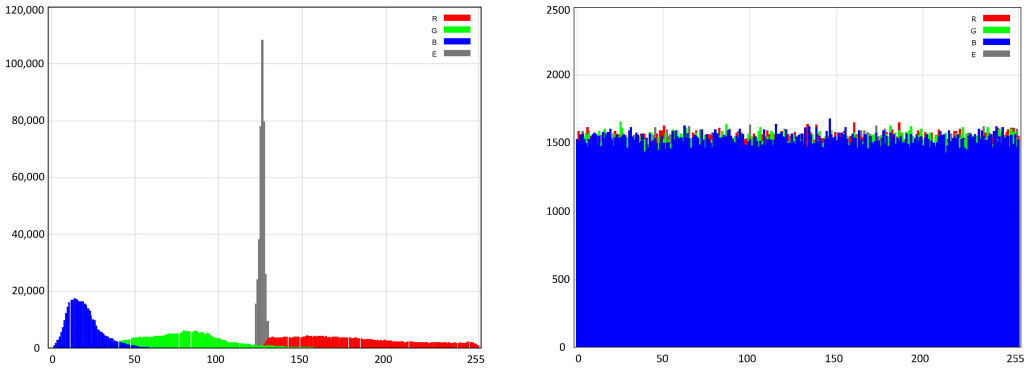
Figure 6. The visual perception of the tone-mapped test image, "memorial," first row, and the test image 8 ("Display1000"), second row. (**a**) the original HDR image in the OpenEXR format; (**b**) the stego images concealing a hidden message in the RGBE format; (**c**) the stego encrypted image; (**d**) the deciphered plaintext image.

Table 5. NIST test results for 2D-SLMM with the initial values $(x_0, y_0) = (0.15874, 0.38797)$.

No	Test Type	*p*-Value	Result	No	Test Type	*p*-Value	Result
1	Frequency	0.931466	Pass	9	Maurer's universal statistical	0.307327	Pass
2	Block frequency	0.926042	Pass	10	Linear complexity	0.917021	Pass
3	Runs	0.652705	Pass	11	Serial	0.880/0.508	Pass
4	Longest run of ones in a block	0.966102	Pass	12	Approximate entropy	0.422853	Pass
5	Binary matrix rank	0.483810	Pass	13	Cummulative sums (forward)	0.969889	Pass
6	Spectral	0.308390	Pass	14	Cummulative sums (reverse)	0.925027	Pass
7	Non-overlapping template matching	0.055807	Pass	15	Random excursions	0.650265	Pass
8	Overlapping template matching	0.477088	Pass	16	Random excursions variant	0.288640	Pass

3.2.3. Histogram Analysis

The histogram contains much statistical information about an image, and the histogram of a plain image contains obvious distribution features. An eligible encryption algorithm should completely eliminate these features. Figure 7 shows histograms of the plaintext image and the ciphered one. Our encryption algorithm makes the distribution more uniform and completely different from the histograms of the plaintext in four channels. No meaningful information can be obtained from the histogram of the stego encrypted HDR image. Therefore, our scheme can effectively prevent statistical attacks.



(**a**) (**b**)

Figure 7. Histograms of the stego HDR RGBE image, "memorial"; (**a**) the histogram of the plain stego image; (**b**) the histogram of the stego encrypted image.

Variance of histogram (VOH) quantitatively reflects the randomness of the image. VOH is calculated by Equation (14), where $Z = \{z_0, z_1, \ldots, z_{255}\}$ denotes the number of counts at each bin in the histogram and μ represents the average value over all bins. The smaller the VOH, the more randomness the test image holds:

$$Var(Z) = \sum_{i=0}^{255} E\left[(z_i - \mu)^2\right] \tag{14}$$

Apart from VOH, we utilize the Chi-square (χ^2) test. When the significance level, α, is set as 0.05 and the degree of freedom is 255; the threshold of the χ^2 test is 293.25. If the χ^2 value of a histogram is under this threshold, the histogram can be considered to perform with uniform distribution statistically.

Table 6 lists VOH and χ^2 values of the "memorial" test image before and after encryption. The results show that encryption reduces the VOH values dramatically, from over several millions to as small as several thousands. In addition, the χ^2 values also show a significant decrease in the encrypted image, from one hundred thousand to several hundreds. Finally, χ^2 values in four channels are below the threshold, thereby all 20 stego encrypted test images pass the χ^2 test.

Table 6. Variance of Histogram (VOH) and χ^2 values for the test image, "memorial".

Channel	VOH		χ^2 Value	
	Plain	Encrypted	Plain	Encrypted
R	2,707,096.750	1653.078	451,182.792	275.513
G	3,589,377.078	1633.258	598,229.513	272.210
B	15,759,698.078	1692.586	2,626,616.346	282.098
E	104,124,933.656	1552.016	17,354,155.610	258.669

To provide more insights, we conducted the χ^2 test for all 20 test images. Figure 8 shows the test results. As can be seen from the figure, all of the χ^2 values in the entire 20 test cases in four channels are below the threshold, shown as a dashed line. Based on the results presented in Table 6 and Figure 8, we conclude that, firstly, VOH and the χ^2 values are greatly reduced in the stego encrypted image in comparison with those appearing in the stego plaintext image. In addition, the χ^2 results confirm the success of passing the χ^2 test. Our proposed algorithm determines a uniform distribution of the encrypted pixels in four channels, offering high security for image encryption.

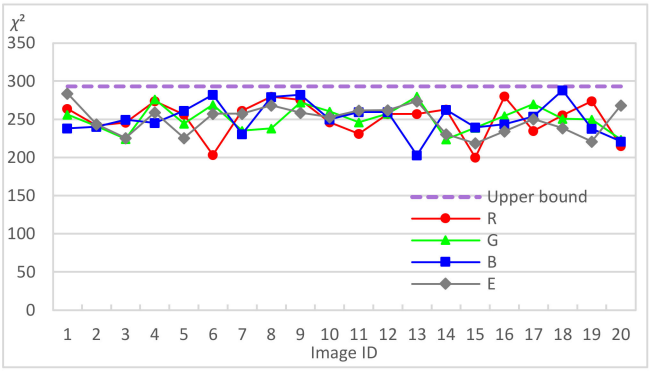


Figure 8. All 20 stego encrypted images in four channels pass the χ^2 test.

3.2.4. Correlation Analysis

In plain images, adjacent pixels are often highly correlated in three directions: horizontal, vertical, and diagonal, which makes it possible to predict current pixels by adjacent pixels. An eligible encryption algorithm should eliminate such correlations. The correlation among pixels can be evaluated by the correlation coefficient in Equation (15):

$$r_{x,y} = \frac{E(x - E(x))E(y - E(y))}{\sqrt{D(x)}\sqrt{D(y)}} \tag{15}$$

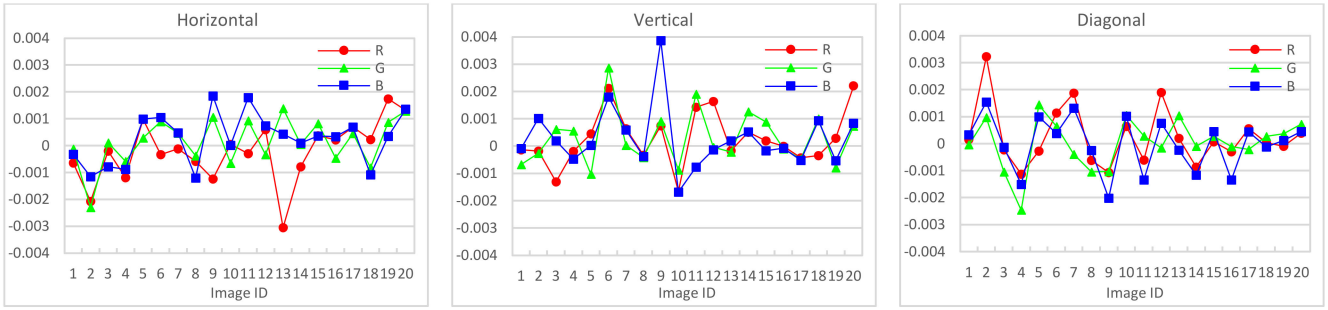
where x and y are values of adjacent pixels. $E(x)$ and $D(y)$ are the expectation and variance of x and y over some pairs of samples, respectively. Finally, the correlation coefficient values $(r_{x,y})$ are between -1 and +1: the closer to zero, the less correlations among chosen pixels. In addition, if the correlation coefficient is a positive number, the variables are directly related. If, on the other hand, the coefficient is a negative number, the variables are inversely related.

We compute the scene-referred color values using Equation (1), which are all floating-point values in the R-, G-, B-channels. Therefore, the analysis is represented in three R-, G-, B-channels. Table 7 shows the correlation coefficients calculated from 5000 pairs of adjacent pixels in three directions: horizontal (H), vertical (V), and diagonal (D) for three channels in the plaintext and encrypted images. The statistics show that the correlation coefficients have been dramatically reduced in the ciphered image, and the values are very close to zero.

Table 7. Adjacent correlation coefficients for the test image "memorial".

Channel	Plaintext Image			Encrypted Image		
	H	***V***	***D***	***H***	***V***	***D***
R	0.869519	0.886546	0.806847	−0.001245	0.000738	−0.001069
G	0.849231	0.871814	0.780838	0.001059	0.000908	−0.001039
B	0.857073	0.877966	0.788200	0.001848	0.003866	−0.002014

The correlation coefficients for all 20 stego test images are shown in Figure 9, where the coefficient coefficients are close to zero from both the positive and negative directions, indicating that our image encryption algorithm is effective in reducing the correlation in three directions.



(**a**) (**b**) (**c**)

Figure 9. Adjacent correlation coefficients are close to zero for all 20 stego test images in the (**a**) horizontal, (**b**) vertical, and (**c**) diagonal directions.

3.2.5. Entropy Analysis

Entropy is a physical quantity to express the degree of uncertainty in a system. Since Shannon [39] introduced the concept to information theory, it is referred to as the Shannon entropy or information entropy.

For an 8-bit grayscale image, Z, the definition of Shannon entropy is shown in Equation (16), where $P(z_i)$ represents the probability of the i-th gray level z_i occurring in the image:

$$H(Z) = -\sum_{i=0}^{255} P(z_i) log_2[P(z_i)] \tag{16}$$

If Z is an ideal random image, $P(z_i) = 1/256$ and $H(Z) = 8$. Therefore, the Shannon entropy for an encrypted 8-bit image is targeted to 8, representing the success of image encryption. Table 8 lists the Shannon entropy for the plaintext image and the encrypted one. The statistics demonstrate that our scheme is so effective that it can produce Shannon entropy for the encrypted RGBE image close to 8 in all channels.

Table 8. Shannon entropy values of the image "memorial" in the plain and encrypted images.

Channel	Plaintext Image	Encrypted Image
Red	6.96112745	7.99949461
Green	6.91334510	7.99950031
Blue	5.46477440	7.99948285
Exponent	2.85423907	7.99949461

Furthermore, Wu et al. [40] proposed the local Shannon Entropy (LSE) to overcome the weakness of the (global) Shannon entropy and introduced a process to conduct the statistical test. For an 8-bit grayscale image, Z, the LSE can be computed in Equation (17), which averages the information entropy on k randomly chosen non-overlapping blocks B_i

each of which contains n pixels. If $(k,\ n) = (30,\ 1936)$, the ideal value of LSE is 7.9025. If the significance level, α, is set as 0.05, the LSE of a ciphered image should be within the range of [7.9019, 7.9030] in order to pass the statistical test:

$$\overline{H_{k,\,n}}(Z) = \sum_{i=1}^{k} \frac{H(B_i)}{k} \tag{17}$$

Figure 10 shows the results of LSE for all 20 stego ciphered images. The figure indicates that only two local Shannon entropy values are not within the range to pass the test, indicating that the pass rate is as high as 97.5%. The global and the local Shannon entropy test confirm that our algorithm generates stego encrypted images exhibiting both global and local randomness; thereby, they are capable of resisting the entropy attacks.

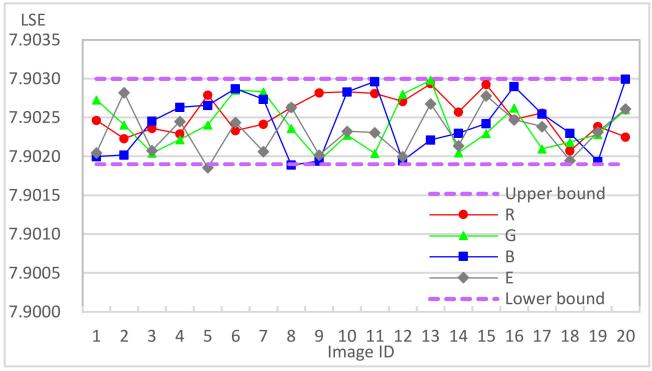


Figure 10. The local Shannon Entropy (LSE) results for all 20 stego ciphered images.

3.2.6. Image Sensitivity

A good algorithm designed for image encryption must be sensitive to any tiny difference between two input images, even though there is only a one-bit difference because a hacker can try to find the relationship between two ciphered images by modifying one bit of the plaintext image. Thus, image sensitivity is also an indicator to show the ability of resisting differential attacks. The number of pixel change rates (NPCR) and the unified averaged changed intensity (UACI) are the two most common quantities used to evaluate the strength of image encryption algorithms with respect to differential attacks [41–43]. NPCR and UACI are defined in Equations (18)–(20), where $I_1(i,j)$ and $I_2(i,j)$ are the pixel values in images I_1 and I_2:

$$D(i,j) = \begin{cases} 0 \ \ if\ I_1(i,j) = I_2(i,j) \\ 1 \ \ if\ I_1(i,j) \neq I_2(i,j) \end{cases} \tag{18}$$

$$NPCR(I_1,\ I_2) = \frac{\sum_{i,j} D(i,\ j)}{H \times V} \times 100\% \tag{19}$$

$$UACI(I_1,\ I_2) = \frac{1}{H \times V}\left(\sum_{i,j} \frac{|I_1(i,j) - I_2(i,j)|}{255}\right) \times 100\% \tag{20}$$

The variance of NPCR and UACI is related to the resolutions of the test images. In addition, it is unclear how high NPCR/UACI is such that the image cipher does actually have a high security level able to resist malicious attacks. To this end, Wu et al. [42] proposed a mathematical model for ideally encrypting images and then they derived expectations and variances of NPCR and UACI used to form statistical hypothesis tests. Their findings indicate that the ideal values of NPCR and UACI are 99.6094 and 33.4635, respectively.

Table 9 reports the results of NPCR and UACI for the test image "memorial". We remark that E-channel has a narrow histogram band in the plaintext image (see Figure 7a). However, our encryption performs so effectively that it shows near uniform distortion after the image encryption, as shown in Figure 7b. Consequently, all NPCR values, including the

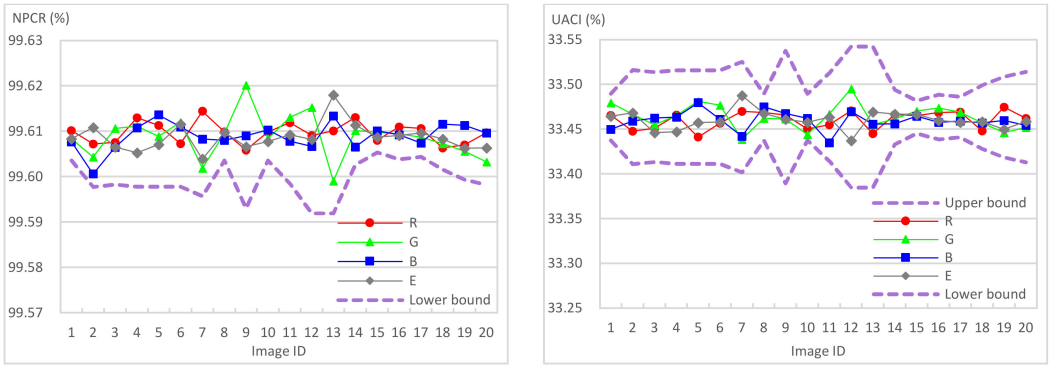
E channel, are close to the ideal value (99.6094), while the UACI statistics are also close to the ideal value (33.4635), reflecting that the stego encrypted image, "memorial", can resist the differential attack.

Table 9. NPCR and UACI values for the stego encrypted test image, "memorial".

Channel	NPCR (%)	UACI (%)
R	99.60586548	33.46571998
G	99.62015788	33.46139127
B	99.60901896	33.46733462
E	99.60657756	33.46112799

We conducted a statistical hypothesis test, suggested in [42] for all 20 stego encrypted images. We first flipped the least significant bit (LSB) of four corner pixels and the center pixel in R-channel, generating five respective stego encrypted images, each of which is one-bit different from its original. We then calculated the NPCR and UACI values for five pairs of images (original vs. the image with a flipping pixel) for analysis.

Figure 11 shows the NPCR and UACI results averaged from five pair of images for 20 stego encrypted images. The NPCR results show that their values are larger than the threshold, thus passing the hypothesis test. The UACI results indicate that their values are within the range formed by the lower bound and upper bound, implying that these images pass the UACI hypothesis test. According to Table 9 and Figure 11, we conclude that our encryption scheme produces good performance in resisting the differential attacks.



(**a**) (**b**)

Figure 11. The results of NPCR and UACI values for statistical hypothesis tests. (**a**) NPCR. (**b**) UACI.

3.2.7. Key Security

Key security can be discussed from the perspectives of key space and key sensitivity. In order to resist the brute force attack, the encryption algorithm must have large key space. In addition, the eligible encryption algorithm needs to be sensitive so that any subtle changes of the keys produce a completely different ciphered image. We discuss the key space and key sensitivity in the following:

- **Key space:** When employing the 2D-SLMM pseudo-random sequence generator, we use three 64-bit double-precision floating-point numbers: h, x_K, y_K, and a 16-bit integer q to discard the first q items to avoid the transient effect. For the encryption algorithm, we adopt five 8-bit integers: $P_{0,R'}$, $P_{0,G'}$, $P_{0,\ B'}$, $P_{0,E'}$, $C_{0,E'}$ as the initial values. Therefore, the key space is 2^{248}, larger than the minimal requirement of 2^{128}, thus capable of resisting a brute force attack.
- **Key sensitivity:** We conducted the key sensitivity test for our algorithm as follows: First, we used two keys, K_1 and K_2, with only one-bit difference. We then encrypted the test image using K_1 and K_2, thus producing EI_{k_1}, shown in Figure 12b and EI_{k_2}, shown in Figure 12d. Next, we decrypted EI_{k_1} using K_1, thus producing the original

image, shown in Figure 12c. In addition, when we decrypted EI_{k_1} using a different key, K_2, the decrypted image produced looks like a noise image containing no useful information, shown in Figure 12f. The NPCR and UACI between EI_{S1} and EI_{S2} are close to the ideal values, 99.6094 and 33.4635. We conclude that our algorithm provides the benefit of strong key sensitivity features.

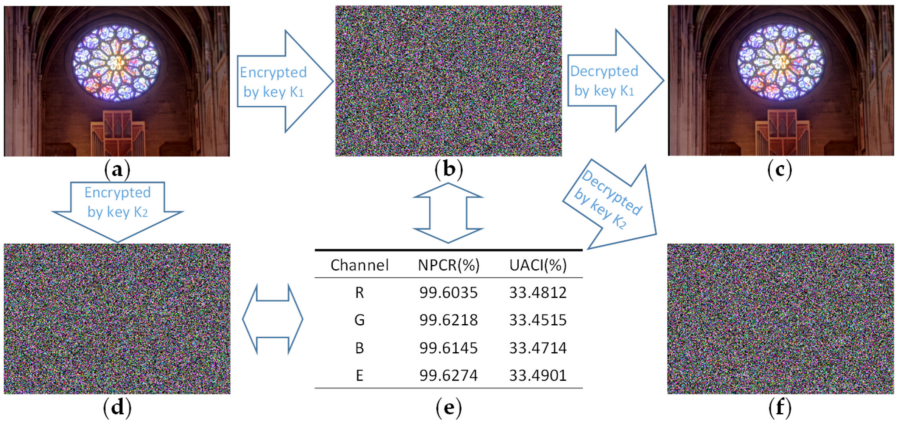


Figure 12. Key sensitivity test using the image, "rosette," where the secret key K_2 is one-bit different from the secret key K_1; (**a**) the original stego image I_S; (**b**) using K_1 to cipher I_S and producing EI_{S1}; (**c**) deciphering EI_{S1} by K_1 to produce the genuine I_S; (**d**) using K_2 to cipher I_S and producing EI_{S2}; (**e**) NPCR and UACI results for the image pair (EI_{S1}, EI_{S2}); (**f**) deciphering EI_{S1} by the incorrect key, K_2, and producing a noisy image.

3.3. A Comparison with the Current State-of-the-Art Works

Table 10 compares our algorithm with the current state-of-the-art works presented for HDR data hiding/HDR image encryption. While most of the schemes were based on RGBE format, only two algorithms adopted the OpenEXR format. The embedding rates for these schemes vary ranging from less than 1.0 bpp to as large as 20.0 bpp. Our algorithm performs better than most competitors except Lin et al.'s work [16]. With regard to the quality of tone-mapped image, our scheme produces a tone-mapped stego image with moderate PSNR (50.49 dB), which is inferior to [16,19] but superior to [17,18]. Note that the PSNR depends not only on the tone mapping algorithm adopted but also on the number of concealed secret messages.

Table 10. A comparison of our scheme with the current state-of-the-art works.

Algorithm	Proposed	[20]	[19]	[18]	[17]	[16]	[13]	[14]	[15]	[12]	[11]
Year	2022	2022	2022	2020	2019	2017	2016	2012	2011	2011	2009
Format	OpenEXR	RGBE	RGBE	RGBE	RGBE	OpenEXR	RGBE	RGBE	RGBE	LogLuv	RGBE
ER (bpp)	7.30~9.29	6.19~7.03	1.07~2.34	0.490~2.292	1.90~2.43	2.433~20.002	0.1391~0.1472	0.1340~0.1373	0.1256~0.1281	26.0	5.04~9.70
PSNR	32.60–51.90	N.A.	61.39–75.66	50.65–51.77	35.96–39.36	45.12–82.32	N.A.	N.A.	N.A.	30.47–37.00	30.00–40.00
SSIM	0.9943–1.0000	N.A.	0.9994–0.9999	0.8542–0.9954	N.A.	0.7572–0.9999	N.A.	N.A.	N.A.	N.A.	N.A.
Encryption	**Yes**	**Yes**	No	No	No	No	No	No	No	No	No
NIST Test	**Yes**	No	No	No	No	No	No	No	No	No	No
Adaptive	**Yes**	No	No	**Yes**	No	**Yes**	No	No	**Yes**	No	**Yes**
Prediction	**Yes**	No	No	No	No	No	No	No	No	No	No
User Req.	**Yes**	No	No	No	No	No	No	No	No	No	No
Constructive	**Yes**	No	No	No	No	No	No	No	No	No	No
Evaluation	Six Metrics	Visual	-	-	-	-	-	-	-	-	-
Security	High	Low	-	-	-	-	-	-	-	-	-

Our algorithm is the first constructive data hiding approach that can adaptively convey secret messages, encrypt the stego HDR image, and offer the prediction ability. Our scheme conceals more secret messages in pixels with low luminance. In addition, the user has flexibility to embed in a high payload with more distortion allowed, or to conceal a low

payload in exchange for high image quality. In this paper, the user demand embedding rate is set to be 7.3 bpp, but our scheme can offer the embedding rate to reach 9.29 bpp, larger than most of our competitors, or alternatively to reduce to 4.3 bpp or even lower. Nevertheless, thanks to the adaptive embedding and the optimal base, the stego image produced by our scheme maintains high image quality. Furthermore, Tsai et al. [20] and our algorithm investigated the HDR image encryption. However, we propose six metrics to evaluate the security analysis completely rather than providing only visual perception. The security analysis confirms that our scheme produces a secure image encryption result to become the current state-of-the-art work.

4. Conclusions and Future Work

In this paper, we proposed a constructive adaptive HDR data hiding method, where a stego HDR image is synthesized during the HDR format conversion. Secret messages are adaptively embedded based on the distribution of the E channel information, so more messages are conveyed in lower-luminance pixels and fewer in higher-luminance areas. Thanks to the optimal base mechanism we propose, our algorithm not only complies with a user's demand, but also generates a stego image with minimal mean squared error. To the best of our knowledge, our algorithm is the first providing prediction and satisfying the user's embedding capacity demand in the HDR image literature. To further protect the stego image from unauthorized user access as well as the hidden secret message, we adopted 2D Sine Logistic modulation map and the sequence produced passed 16 randomness tests in the NIST SP 800-22 test suite, confirming that it has better hyperchaotic behavior to cipher the stego HDR image. We introduced a random permutation technique able to fully shuffle the pixel contents, thus achieving the bit-level permutation image ciphering. We adopted six metrics to thoroughly and comprehensively evaluate the security of the stego ciphered HDR RGBE image. Our scheme offers 18% to 32% larger embedding rate than the current state-of-the-art schemes' results without degrading the quality of stego image. The security evaluation confirms that our scheme provides high security that is superior to the competitors.

Our future work is to extend the current algorithm to provide the reversibility, able to restore the original HDR image after message extraction, and to improve our algorithm by taking into consideration human visual sensitivity for message embedding.

Author Contributions: Conceptualization, C.-F.L. and C.-M.W.; methodology, C.-F.L.; software, C.-F.L.; validation, C.-F.L.; writing—original draft preparation, C.-F.L. and C.-M.W.; writing—review and editing, C.-F.L. and C.-M.W.; project administration, C.-M.W. and W.L.; Supervision, W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available in a publicly accessible repository.

Acknowledgments: This work is supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 107-2221-E-005-069, 108-2221-E-005-051, MOST 109-2221-E-005-062, MOST 110-2221-E-005-069, and NSC-111-2221-E-005-076.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Reinhard, E.; Ward, G.; Pattanaik, S.; Debevec, P.; Heidrich, W.; Myszkowski, K. *High Dynamic Range Imaging, Acquisition, Display, and Image-Based Lighting*, 2nd ed.; Morgan Kaufmann: Burlington, VT, USA, 2010.
2. Kim, M.; Kautz, J. Consistent tone reproduction. In Proceedings of the 10th IASTED International Conference on Computer Graphics and Imaging, (CGIM 2008), Innsbruck, Austria, 13–15 February 2008; pp. 152–159.
3. Ward, G.J. The RADIANCE lighting simulation and rendering system. In Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, Orlando, FL, USA, 24–29 July 1994; pp. 459–472. [CrossRef]
4. Larson, G.W. LogLuv encoding for full-gamut, high-dynamic range images. *J. Graph. Tools* **1998**, *3*, 15–31. [CrossRef]

5. OpenEXR. Available online: https://www.openexr.com (accessed on 29 September 2022).
6. Cerad-Company, X.; Parraga, C.A.; Otazu, X. Which tone-mapping operator is the best? A comparative study of perceptual quality. *J. Opt. Soc. Am. A* **2018**, *35*, 626–638. [CrossRef] [PubMed]
7. Kadhim, I.J.; Premaratne, P.; Vial, P.J.; Halloran, B. Comprehensive survey of image steganography: Techniques, evaluations, and trends in future research. *Neurocomputing* **2019**, *335*, 299–326. [CrossRef]
8. Kacar, S.; Konyar, M.Z.; Cavusoglu, U. 4D chaotic system-based secure data hiding method to improve robustness and embedding capacity of videos. *J. Inf. Secur. Appl.* **2022**, *71*, 103369. [CrossRef]
9. Wu, H.-Y.; Chen, L.-H.; Ching, Y.-T. Block-based steganography method using optimal selection to reach high efficiency and capacity for palette images. *Appl. Sci.* **2020**, *10*, 7820. [CrossRef]
10. Chen, Y.-H.; Chang, C.-C.; Lin, C.-C.; Wang, Z.-M. An adaptive reversible data hiding scheme using AMBTC and quantization level difference. *Appl. Sci.* **2021**, *11*, 635. [CrossRef]
11. Cheng, Y.-M.; Wang, C.-M. A novel approach to steganography in high-dynamic-range images. *IEEE Multimed.* **2009**, *16*, 70–80. [CrossRef]
12. Li, M.-T.; Huang, N.-C.; Wang, C.-M. A data hiding scheme for high dynamic range images. *Int. J. Innov. Comput. Inf. Control* **2011**, *7*, 2021–2035.
13. Chang, C.-C.; Nguyen, T.-S.; Lin, C.-C. A new distortion-free data embedding scheme for high-dynamic range images. *Multimed. Tools Appl.* **2016**, *75*, 145–163. [CrossRef]
14. Wang, Z.-H.; Lin, T.-Y.; Chang, C.-C.; Lin, C.-C. A novel distortion-free data hiding scheme for high dynamic range images. In Proceedings of the Fourth International Conference on Digital Home, Guangzhou, China, 25 November 2012; pp. 33–38. [CrossRef]
15. Yu, C.-M.; Wu, K.-C.; Wang, C.-M. A distortion-free data hiding scheme for high dynamic range images. *Displays* **2011**, *32*, 225–236. [CrossRef]
16. Lin, Y.-T.; Wang, C.-M.; Chen, W.-S.; Lin, F.-P.; Lin, W. A novel data hiding algorithm for high dynamic range images. *IEEE Trans. Multimed.* **2017**, *19*, 196–211. [CrossRef]
17. He, X.; Zhang, W.; Zhang, H.; Ma, L.; Li, Y. Reversible data hiding for high dynamic range images using edge information. *Multimed. Tools Appl.* **2019**, *78*, 29137–29160. [CrossRef]
18. Gao, X.; Pan, Z.; Gao, E.; Fan, G. Reversible data hiding for high dynamic range images using two-dimensional prediction-error histogram of the second time prediction. *Signal Process.* **2020**, *173*, 107579. [CrossRef]
19. Tsai, Y.-Y.; Liu, H.-L.; Ying, C.-Y. Applying homogeneity index modification to high-capacity high-dynamic-range image authentication with distortion tolerance. *Multimed. Tools Appl.* **2022**, *81*, 24957–24976. [CrossRef]
20. Tsai, Y.-Y.; Liu, H.-L.; Kuo, P.-L.; Chan, C.-S. Extending multi-MSB prediction and Huffman coding for reversible data hiding in encrypted HDR Images. *IEEE Access* **2022**, *10*, 49347–49358. [CrossRef]
21. Wu, K.-C.; Wang, C.-M. Steganography using reversible texture synthesis. *IEEE Trans. Image Process.* **2014**, *24*, 130–139. [CrossRef]
22. Hsieh, K.-S.; Wang, C.-M. Constructive image steganography using example-based weighted color transfer. *J. Inf. Secur. Appl.* **2022**, *65*, 103126. [CrossRef]
23. Singh, K.N.; Singh, A.K. Towards integrating image encryption with compression: A survey. *ACM Trans. Multimed. Comput. Commun. Appl.* **2022**, *18*, 89. [CrossRef]
24. Zia, U.; McCartney, M.; Scotney, B.; Martinez, J.; Abu Tair, M.; Memon, J.; Sajjad, A. Survey on image encryption techniques using chaotic maps in spatial, transform and spatiotemporal domains. *Int. J. Inf. Secur.* **2022**, *21*, 917–935. [CrossRef]
25. Singh, M.; Singh, A.K. A comprehensive survey on encryption techniques for digital images. *Multimed Tools Appl.* **2022**. [CrossRef]
26. Kumari, M.; Gupta, S.; Sardana, P. A survey of image encryption algorithms. *3D Res.* **2017**, *8*, 37. [CrossRef]
27. Yan, J.-Y.; Chen, T.-H.; Lin, C.-H. Encryption in high dynamic range images for RGBE format. In Proceedings of the Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Beijing, China, 16–18 October 2013; pp. 493–496. [CrossRef]
28. Lin, K.-S.; Chen, T.-H.; Lin, C.-H.; Chang, S.-S. A tailor-made encryption scheme for high-dynamic range images. *Adv. Intell. Syst. Comput.* **2014**, *238*, 183–192. [CrossRef]
29. Chen, T.-H.; Chang, S.-S. Image encryption on HDR images for OpenEXR Format. *Int. J. Eng. Sci.* **2014**, *4*, 19–23.
30. Chen, T.-H.; Yan, J.-Y. Commutative encryption and authentication for OpenEXR high dynamic range images. *Multimed. Tools Appl.* **2021**, *80*, 27807–27828. [CrossRef]
31. Li, W.; Yan, A.; Zhang, H. Novel multiple-image encryption scheme based on coherent beam combining and equal modulus decomposition. *Appl. Sci.* **2021**, *11*, 9310. [CrossRef]
32. Liu, Z.; Guo, Q.; Xu, L.; Ahmad, M.A.; Liu, S. Double image encryption by using iterative random binary encoding in gyrator domains. *Opt. Express* **2010**, *18*, 12033–12043. [CrossRef]
33. Hua, Z.; Zhou, Y.; Pun, C.-M.; Chen, C.L. Philip. 2D Sine Logistic modulation map for image encryption. *Inf. Sci.* **2015**, *297*, 80–94. [CrossRef]
34. *IEEE STD 754-2019*; IEEE Standard for Floating-Point Arithmetic. IEEE Computer Society: Piscataway, NJ, USA, 2019; pp. 1–84. [CrossRef]
35. Durstenfeld, R. Algorithm 235: Random permutation. *Commun. ACM* **1964**, *7*, 420. [CrossRef]

36. High Dynamic Range Image Examples. Available online: http://www.anyhere.com/gward/hdrenc/pages/originals.html (accessed on 28 October 2022).
37. Mantiuk, R.; Daly, S.; Kerofsky, L. Display adaptive tone mapping. *ACM Trans. Graph.* **2008**, *27*, 110. [CrossRef]
38. Bassham, L.E.; Rukhin, A.L.; Soto, J.; Nechvatal, J.R.; Smid, M.E.; Barker, E.B.; Leigh, S.D.; Levenson, M.; Vangel, M.; Banks, D.L.; et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; Technical Report; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2010.
39. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
40. Wu, Y.; Zhou, Y.; Saveriades, G.; Agaian, S.; Noonan, J.P.; Natarajan, P. Local Shannon entropy measure with statistical tests for image randomness. *Inf. Sci.* **2013**, *222*, 323–342. [CrossRef]
41. Chen, G.; Mao, Y.; Chui, C. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos Solitons Fractals* **2004**, *21*, 749–761. [CrossRef]
42. Wu, Y.; Noonan, J.P.; Agaian, S. NPCR and UACI randomness tests for image encryption. *Cyber J. Multidisci. J. Sci. Technol. J. Select. Areas Telecommun.* **2011**, *2*, 31–38.
43. Zhang, Y. Statistical test criteria for sensitivity indexes of image cryptosystems. *Inf. Sci.* **2021**, *550*, 313–328. [CrossRef]