applied sciences

MDPI

Article

Advancement of the DRPE Encryption Algorithm for Phase CGHs by Random Pixel Shuffling

Alfonso Blesa [1,*,†] and **Francisco J. Serón** [2,†]

[1] Department of Electronics and Communication, EUPT, Universidad de Zaragoza, 44003 Teruel, Spain

[2] Department of Computer Science, EINA, Universidad de Zaragoza, 50008 Zaragoza, Spain; seron@unizar.es

* Correspondence: ablesa@unizar.es

† These authors contributed equally to this work.

Abstract: This work presents an optical encryption process for various types of information related to 3D worlds (scenes) or 2D images, utilizing Computer-Generated Holograms (CGHs). It also introduces a modification to the Dual Random Phase Encoding (DRPE) encryption algorithm by incorporating pixel shuffling. This proposal enables the use of either a single key for both pixel shuffling and phase mask definition or two independent keys. The latter option is particularly advantageous in applications that require the involvement of two independent agents to retrieve the original plaintext. The dimension of the CGHs determines the size of the keys based on the random generation of values by cryptographically secure algorithms, so the use of arithmetic encryption is proposed for data compression. However, this proposal allows the use of other algorithms described in the literature to generate the shuffle and phase matrices. The complete workflow is described starting from the synthesis of a 3D scene, defined by a mesh of triangles with shape and appearance modeling, or 2D images of any level of geometric or visual complexity using computer graphics; its storage in a CGH, the encryption and decryption process, and finally, the results obtained in the laboratory and by simulation are shown. The similarity between different encryption levels is measured by the Pearson Coefficient to evaluate the results obtained.

Keywords: Computer-Generated Hologram; optical encryption; double random phase encoding; pixel permutation; secure analysis; 3D computer graphics; random pixel shuffling

check for updates

Academic Editors: Amalia Martínez-García and Miguel León-Rodríguez

Received: 7 March 2025
Revised: 1 April 2025
Accepted: 6 April 2025
Published: 9 April 2025

Citation: Blesa, A.; Serón, F. J. Advancement of the DRPE Encryption Algorithm for Phase CGHs by Random Pixel Shuffling. *Appl. Sci.* **2025**, *15*, 4120. https://doi.org/10.3390/app15084120

1. Introduction

Considering that the visual sense has the highest bandwidth among all human senses, an image is one of the most natural ways to receive large amounts of information. For this reason, science and technology have always shown a particular interest in understanding and modeling all aspects related to image formation, modification, storage, and transmission. Optics is the discipline that embodies this knowledge, accumulated over centuries, and provides a model for the propagation of information and energy through electromagnetic waves (in the visible spectrum) [1].

However, it is in recent decades that photonic and optical techniques have emerged as powerful tools for processing the information contained in an image, especially as previously developed models have been implemented into computer-executable algorithms. Among all these technologies, Computer-Generated Holograms (CGHs) have proven to be a viable option for massive information storage, typically associated with 2D images or 3D scenes. This process requires a significant amount of computational resources, which, in some cases, exceed the capabilities of current computers if synthesis is to be performed within acceptable time frames.

The creation, manipulation, storage, and encryption of images (initially in black and white and later in color) have been the subject of interest and study since the beginning of all civilizations. However, it is nowadays, with the evolution of technologies linked to the digital era, that the need to protect and securely transmit this type of information has become more evident. Image encryption is a dynamic discipline that currently allows for multiple approaches [2,3].

Similarly, the information contained in the CGH must also be securely stored and transmitted. A CGH needs more information for its synthesis. From an optical point of view, it is necessary to know the contribution of the amplitudes and phases of the elements of the scene for each element (pixel) of the CGH, which means that characteristics typical of traditional images (e.g., correlations between nearby pixels or information redundancy) usually do not occur. Consequently, specialized encryption techniques have been developed in parallel to safeguard the integrity of CGHs and prevent unauthorized access [4,5].

We must keep in mind that a CGH can store images, but it can also store more complex data structures, for example, 3D scenes. The differential characteristic of working with 3D virtual scenes defined by triangle meshes is that they allow for modeling the aspects of shape and appearance, offering the possibility of obtaining different views of the scene. However, to retrieve this information (stored in a file format to be determined), it is necessary to know parameters such as the geometry of the scene, its location with respect to the observer, the resolution used, and the illumination used.

In this paper, we focus on the use of CGH as a solvent option that allows for saving this information in an image-like format (i.e., saving the phase pattern in a png file for later presentation to display devices such as SLMs). This approach allows for easily integrating encryption and decryption processes well known in the field of digital imaging.

In this context, our work consists of the following steps and contributions:

- From a complex virtual scene, designed by means of computer graphics (CGs), we generate a CGH in which relevant scene information such as luminosity, depth, and occlusion is stored.
- To encrypt this information, we present an evolution of the DRPE algorithm by incorporating random pixel shuffling. This significantly increases the robustness of the encryption against brute force attacks.
- Two keys are defined: one for the pixel shuffling process and one for the random mask phase modification process.
- The proposal allows these keys to be generated from a single key or to be independent in order to improve the level of encryption security.
- The decrypted scene is obtained by simulation or in the laboratory.

In Section 2, various image encryption techniques are described, with a particular focus on the DRPE algorithm. In Section 3, we describe the complete workflow, starting from the generation of a 3D scene using computer graphics and its storage in a CGH, followed by the encryption and decryption processes, and concluding with its final visualization.

This work places particular emphasis on proposals related to the encryption phase. The novelty of this approach lies in the fact that it allows, if desired, the use of a single key for both the pixel permutation process and the phase modification, generating the encrypted CGH, which serves as the ciphertext. Section 4 presents results and discussion and performance analysis of our methods. Finally, Section 5 highlights the main conclusions of this work.

2. Preliminaries

2.1. Number and Text Encryption Methods

If we understand a 3D scene as a collection of numbers defining its geometry, lighting, appearance, or position (among other characteristics) stored in a data file, we can consider well-established algorithms for encrypting numerical sequences in the literature. These include symmetric encryption, such as the well-known Advanced Encryption Standard (AES) [6,7], which requires a shared secret key for both encryption and decryption. Additionally, asymmetric encryption employs key pairs (public and private), as seen in algorithms like RSA [8], allowing encrypted data to be shared in a way that only the recipient with the private key can decrypt.

Advanced encryption algorithms are tailored to specific applications, such as stream ciphers like RC4 [9], which are used for encrypting streaming data (e.g., audio or video). Advanced block ciphers, including AES, Twofish, and Serpent, are highly resistant to attacks due to their use of various substitution, permutation, and algebraic techniques. Homomorphic encryption [10] enables computations to be performed on encrypted data without exposing it, eliminating the need for decryption (e.g., Microsoft SEAL [11]). Elliptic Curve Cryptography (ECC) [12] provides high-security encryption while requiring significantly shorter key lengths compared with RSA, making it particularly suitable for resource-constrained devices such as smartphones. Post-quantum cryptographic algorithms [13], such as Kyber [14], NTRU [15], and Dilithium [16], are specifically designed to withstand attacks from quantum computing and are essential for securing data that must remain protected over the long term.

2.2. Image Encryption Methods

Image encryption is based on converting the intensity matrices for each channel into others with indecipherable patterns for unauthorized access. There are multiple proposals to achieve this: chaotic systems [17] offer high security, computational efficiency, and simplicity of implementation: they have features as ergodicity, unpredictability, randomness, and extreme sensitivity to initial conditions and key accuracy, but they show some weaknesses related with high numerical precision or security [18]. There is intensive academic work to address these limitations [19–21].

Currently, there are also proposals based on quantum computation [22,23] that provide robust security against quantum attacks, but face challenges due to the lack of mature quantum infrastructure and its heavy reliance on specialized hardware. Operations based on DNA computing [24,25] offer parallel processing and the ability to handle large amounts of data, but come at the cost of complexity of implementation and the need for specialized laboratory infrastructure.

Image encryption using neural networks [26–28] allows the generation of complex (non-linear) keys and is resistant to classical attacks (such as pattern analysis). They are systems that require a high computational cost in the training process, and this must be directed to specific applications.

Optical encryption using optical transformations [4,29–31] leverages properties such as diffraction and Fourier transforms to securely encode images, offering high-speed processing and resistance to traditional computational attacks, but is limited by the need for specialized hardware and environmental sensitivity .

CGH Encryption Methods: DRPE Encryption

A widely used optical encryption method for image applications is Double Random Phase Encoding (DRPE) [32]. In DRPE, the input image is represented by the amplitude of light, which is modulated by a random phase before being transformed by a Fourier transform. A second random phase mask is then applied in the frequency domain as an encryption key. The resulting complex amplitude is considered the encrypted information. An evolution of this approach is optical encryption using digital holography [33]. This technique can also be modeled using algebraic procedures, such as vector-matrix multiplications [34].

The double-phase encryption technique has been successfully tested using both static and dynamic keys. However, this method presents certain vulnerabilities to attacks, as recognized in the literature. If the process is purely linear, there is a possibility of breaking the encryption key [35]. To enhance DRPE, various improvements have been proposed. One approach involves modifying pixels using chaotic functions [36], while another leverages deep learning techniques [37].

Traditionally, the phase masks proposed for DRPE have aimed to ensure uniformity in both the amplitude of the hologram and the phase distribution of the encrypted CGH to make unauthorized decryption more challenging. Various photosensitive materials (e.g., photorefractive media) have been used for their implementation [38]. Currently, with advancements in spatial light modulator (SLM) technology, diffractive elements computed digitally (such as CGHs) and phase masks can be directly applied to a light beam in the laboratory. This enables both the encrypted data and the encryption keys to be dynamic.

3. From the Synthesis of a 3D Scene to Its Safe Reception: Process Description

A flowchart of the proposed process is shown in Figure 1. It illustrates the different stages required to securely transmit a scene that has been previously modeled using computer graphics techniques (Figure 1a), stored in a CGH (Figure 1b), encrypted (Figure 1c), decrypted (Figure 1d), and visualized (Figure 1e). Algorithms and procedures are shown in blue, results that can be visualized on an SLM or simulated via a computer are in orange, vectors used to manipulate information during the encryption process are in green, and the key used for encryption or decryption is in red.

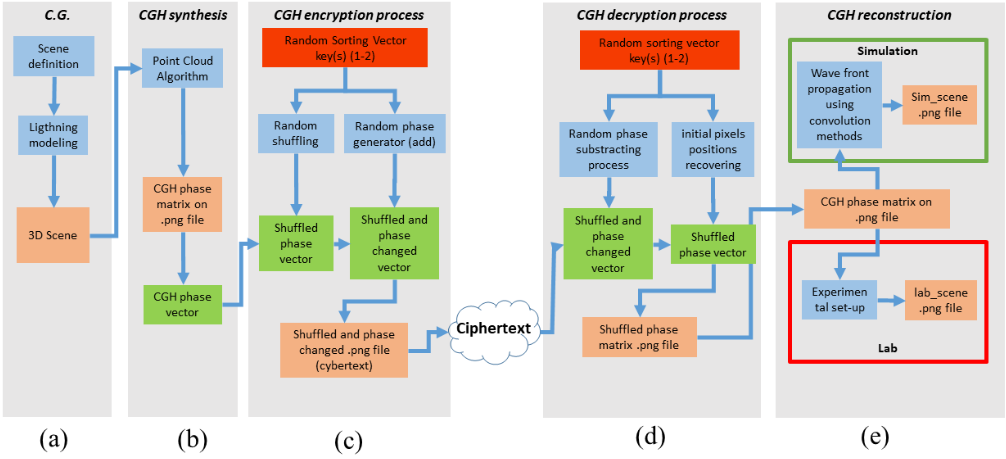

C.G.
Scene definition
Ligthning modeling
3D Scene
CGH synthesis
Point Cloud Algorithm
CGH phase matrix on .png file
CGH phase vector
CGH encryption process
Random Sorting Vector key(s) (1-2)
Random shuffling
Random phase generator (add)
Shuffled phase vector
Shuffled and phase changed vector
Shuffled and phase changed .png file (cybertext)
Ciphertext
CGH decryption process
Random sorting vector key(s) (1-2)
Random phase substracting process
initial pixels positions recovering
Shuffled and phase changed vector
Shuffled phase vector
Shuffled phase matrix .png file
CGH reconstruction
Simulation
Wave front propagation using convolution methods
Sim_scene .png file
CGH phase matrix on .png file
Experimen tal set-up
lab_scene .png file
Lab
(a)
(b)
(c)
(d)
(e)

Figure 1. Flowchart of the method for generating a CGH from a virtual scene, encryption, decryption, and visualization: (**a**) generation of the synthetic image using path tracing techniques; (**b**) CGH synthesis; (**c**) CGH encryption in two steps: pixel shuffling and random phase modification; (**d**) decryption; and (**e**) visualization, both through simulation and in the laboratory.

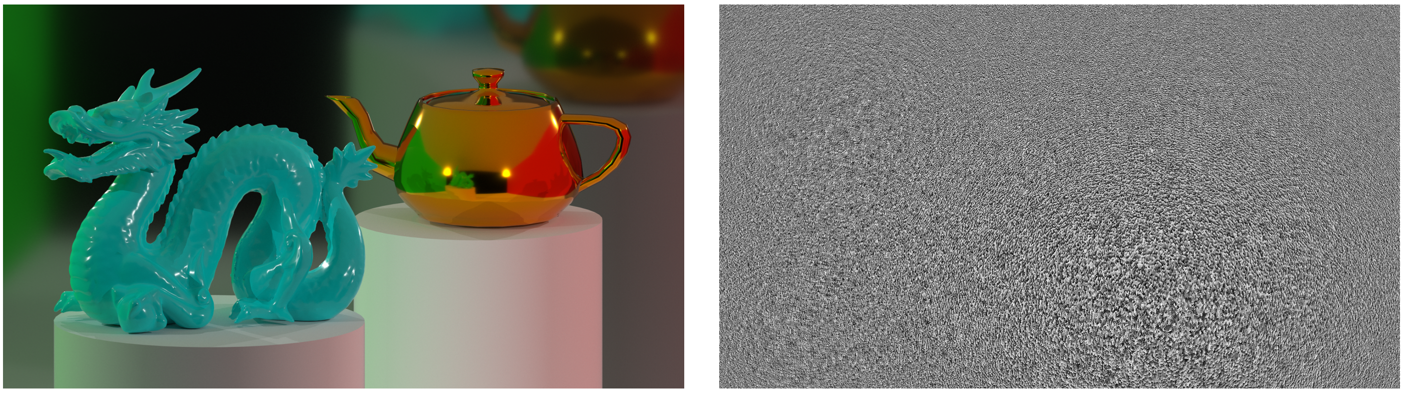
3.1. Scene and CGH Synthesis

To illustrate the encryption algorithm used, a 3D scene has been designed with well-known objects in the field of computer graphics: a dragon and a teapot contained within a Cornell Box (Figure 2a).

Using the ray tracing technique "path tracing" [39], a CGH is computed and stored in a matrix of $N \times M = 1920 \times 1080$ elements. The phase information can be converted into a PNG-format image, which can then be displayed on an SLM in our case, the Pluto Holoeye [40], with a pixel size of 8 µm.

The CGH calculation already inherently includes the first phase mask described in the DRPE algorithm, and the presented procedure allows for encoding both 2D and 3D scenes. This phase corresponds to Figure 1b.

Figure 2a shows the proposed scene, and Figure 2b displays the phase of the computed CGH in PNG format. The scene window matches that of the SLM (8 mm × 15 mm). The detailed process of scene synthesis and CGH generation can be found in [41].



(**a**) 3D Scene.

(**b**) Phase CGH.

Figure 2. Initial scene. (**a**) is the designed scene. (**b**) represents the phase information of the CGH encoded in a PNG image for subsequent transmission to an SLM.

3.2. Encryption of the CGH

Figure 3a shows a flowchart for key generation and CGH encryption. The proposed method involves applying the DRPE algorithm for phase modification and pixel shuffling of the CGH. We will see that pixel shuffling creates a combinatorial explosion that is difficult to attack by brute force. Additionally, phase modification adds an extra layer of encryption to the information stored in the CGH. This stage corresponds to Figure 1c.

To select a truly random key, a cryptographically secure pseudorandom number generator (CSPRNG) must be used. In this work, the `random` function from MATLAB's System.Security.Cryptography library has been used.

These numbers are used as a random index to feed the Fisher–Yates algorithm [42], which is a classic algorithm for randomly and securely shuffling a set of elements. It works by traversing the elements of a vector in reverse order and generating a random index j (obtained by CSPRNG)) for each element i, such that $0 \leq j \leq i$, with i being the index of the element being modified at each step. Finally, the elements i and j are swapped in position.

This information is stored in the Random Sorting Vector (RSV) of size $Z = M \times N$ (Figure 3a). The encryption process uses the phase matrix of the CGH, stored in a PNG file, to convert it into a CGH-PV of size $Z = M \times N$. Using the shuffling RSV, the positions of the CGH-PV are changed, resulting in the Shuffled Phase Vector (SPV).

From the RSV, a phase mask vector can be obtained through a truncation operation.

$$PRV[i] = RSV[i]\%256 \quad \forall \quad i \in \{1, \ldots, Z\} \tag{1}$$

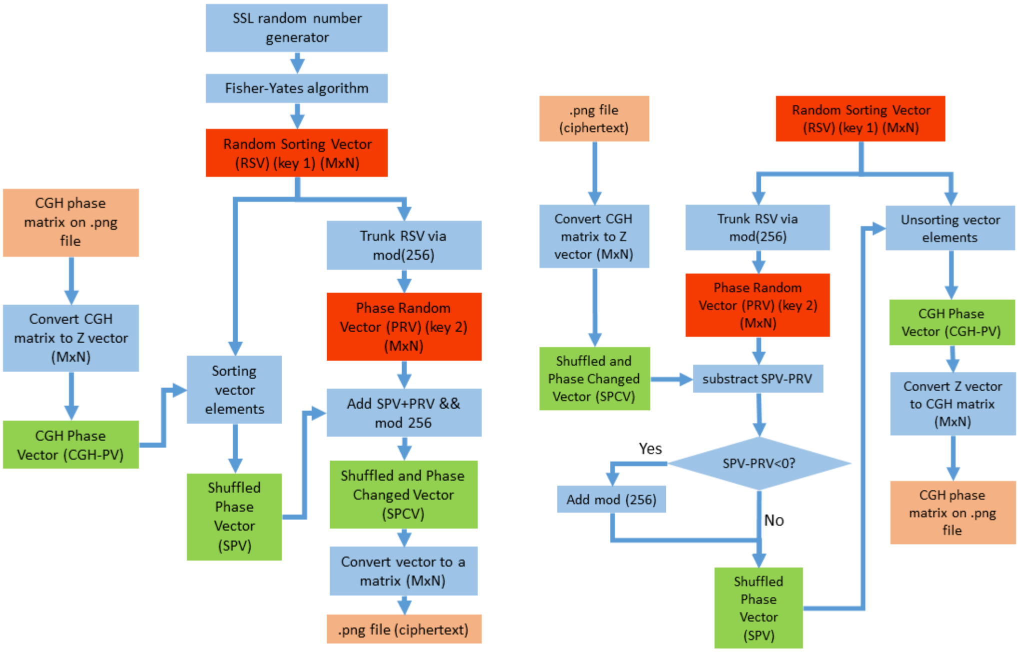
It is necessary to ensure that the values are within range through a modulo operation.

Depending on the required security level, the phase mask can be generated from the shuffling vector through a modulo operation on this vector, both in the encryption phase and in the decryption phase, or with a separate procedure.

This vector is added to the SPV to modify the phase pattern, resulting in the Shuffled and Phase Changed Vector (SPCV).

$$SPCV[i] = (SPV[i] + PRV[i])\%256 \quad \forall \quad i \in \{1, \ldots, Z\} \tag{2}$$

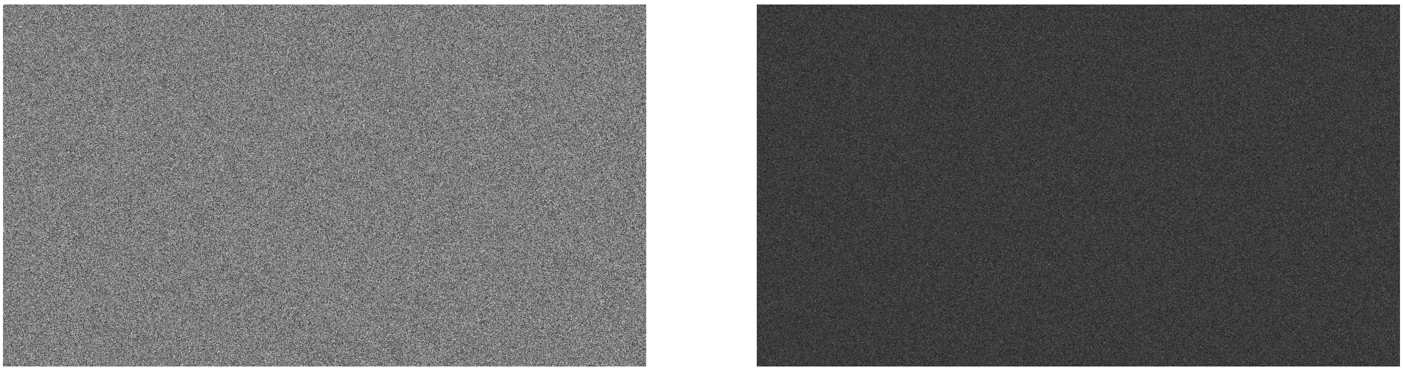
The SPCV is reordered to recover a matrix of size $M \times N$, which will be the ciphertext to be sent to a receiver.


SSL random number generator
Fisher-Yates algorithm
Random Sorting Vector (RSV) (key 1) (MxN)
CGH phase matrix on .png file
Trunk RSV via mod(256)
Convert CGH matrix to Z vector (MxN)
Phase Random Vector (PRV) (key 2) (MxN)
Sorting vector elements
Add SPV+PRV && mod 256
CGH Phase Vector (CGH-PV)
Shuffled and Phase Changed Vector (SPCV)
Shuffled Phase Vector (SPV)
Convert vector to a matrix (MxN)
.png file (ciphertext)
.png file (ciphertext)
Random Sorting Vector (RSV) (key 1) (MxN)
Convert CGH matrix to Z vector (MxN)
Trunk RSV via mod(256)
Unsorting vector elements
Phase Random Vector (PRV) (key 2) (MxN)
CGH Phase Vector (CGH-PV)
Shuffled and Phase Changed Vector (SPCV)
substract SPV-PRV
Convert Z vector to CGH matrix (MxN)
Yes
SPV-PRV<0?
Add mod (256)
No
CGH phase matrix on .png file
Shuffled Phase Vector (SPV)

(**a**) Encryption workflow. (**b**) Decryption workflow.

Figure 3. Flowcharts of the CGH encryption and decryption process. In orange, the files that can be visualized on an SLM are indicated; in red, the keys generated for shuffling and creating the phase mask; in blue, the algorithms or procedures used; and in green, the vectors generated from the previous files and algorithms.

The result of the encrypted CGH obtained (the ciphertext) is shown in Figure 4a, and the image obtained with an incorrect key is shown in Figure 4b. As we will see in Section 4, the histogram of Figure 4b is nearly uniform.



(**a**) Encrypted CGH. (**b**) Image decrypted with wrong key.

Figure 4. Encrypted CGH. (**a**) shows the phase distribution after the encryption operations have been performed. (**b**) is the image obtained from the CGH without using a key or with a wrong key.

3.3. Decryption and Visualization of the Scene Stored in the CGH

Decryption (Figures 1d and 3b) involves undoing the steps from the previous section. In the decryption process, this random phase contribution is subtracted from the SPCV to recover the SPV, as follows:

$$SPV[i] = SPVC[i] - PRV[i] \quad \forall \quad i \in \{1, \ldots, Z\} \tag{3}$$

The original order is then restored to obtain the CGH-PV using the indices stored in the RSV (Figure 3b). Negative values resulting from this subtraction are adjusted by adding the modulo to ensure that they remain within the allowed range of phase values.

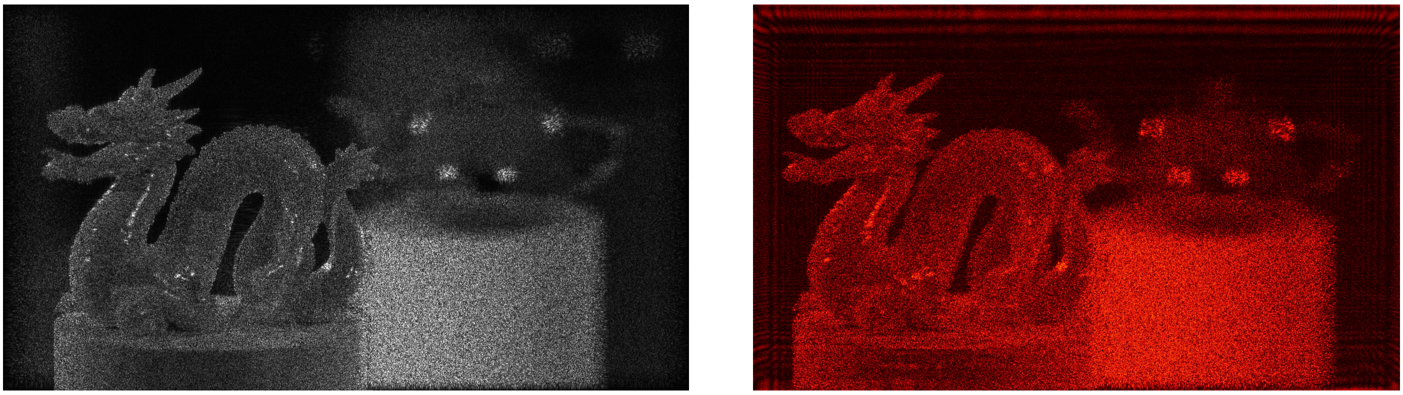
From this vector, the phase matrix of the CGH is obtained to recover an image file (png format). With the correctly decrypted CGH, the original scene can be recovered. This step corresponds to Figure 1e. Figure 5 shows the results for phase holograms encoded with integer values [0–255] corresponding to images in PNG format. In our case, since the phase information is encoded with integer numbers, no information is lost.

3.4. Visualization

The original scene can be recovered either by simulation (using convolution algorithms such as the angular spectrum [1]) (see Figure 5a) or in the laboratory (Figure 5b).

The experimental setup for obtaining the reconstructed scene in the laboratory is shown in Figure 6. The CGH was computed for a HeNe laser source (λ = 0.633 µm). The SLM used is a Pluto from Holoeye [40]. The plane where the camera's CCD is located must coincide with one of the planes in which the real image of the scene was computed. In the case of Figure 5b, this corresponds to the position of the dragon's head.

In both cases (simulation and lab), the behavior of a plane wavefront disturbed by the phase information of the CGH at each pixel is visualized in a specific plane (which coincides with the position of the dragon's head in the 3D scene). To obtain the image in Figure 5b, a CCD from a lensless camera is placed in this plane.



(**a**) Simulation. (**b**) Lab.

Figure 5. Decrypted image from the phase information of the CGH using the right key: (**a**) results obtained by simulation (propagation using the angular spectrum convolution technique); (**b**) image obtained in the laboratory. The plane of focus coincides with the dragon's head, so the teapot is out of focus.

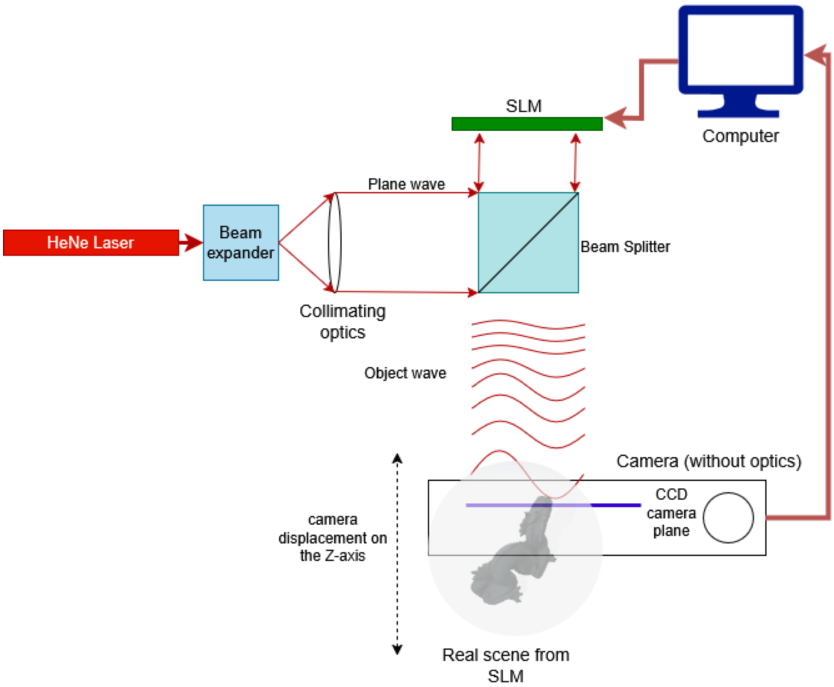


Figure 6. Basic scheme for capturing the scene in the laboratory.

4. Results Discussion and Performance Analysis

The generation of a 3D scene and its subsequent storage, encryption, and later retrieval through simulation or in the laboratory allow us to conduct the following general analysis. To ensure the robustness of an encryption algorithm, it must be capable of transforming the image into an unrecognizable form equivalent to random noise [43]. Therefore, the proposed procedure is robust to different types of attacks owing to the following:

- CGH Storage: The 3D scene is stored within a CGH. Vulnerability to Statistical Analysis is avoided since methods such as intensity histograms cannot reveal patterns or facilitate partial image reconstruction. It is also resistant to partial transformations, as modifying a subset of pixels in the CGH does not compromise the integrity of the stored information.
- Pixel Shuffling: As will be shown next, pixel shuffling can be robust against brute force attacks when the number of possible pixel permutations is sufficiently large.
- Random Phase Modification: This adds an additional layer of complexity to the encryption, as it modifies the phase disturbance that each pixel of the CGH presents to the reconstruction wavefront. It also alters the phase histogram of the ciphertext, further complicating attacks using statistical tools.

4.1. Visual Assessment

Our method enables the storage, encryption, and decryption of multichannel scenes and images. Computer graphics allows for the generation of any multichannel image, which can be stored in a color CGH [44]. This study presents results for a single wavelength. To achieve this, we converted Figure 2a to luminance values to facilitate the comparison between simulation and laboratory results. Nevertheless, the obtained results are extrapolable to color CGHs generated from the multichannel scene.

The quality of the final image is usually affected by several factors: Working with only the phase and encoding it with integer numbers within the allowed range impacts the final scene quality. The dynamic range or the pixel size of the SLM (real or simulated) also directly affects the final quality. Additionally, speckle phenomena always appear, which are not attributable to the encryption technique. To improve the final image, the process can be iterated by generating multiple CGHs and applying statistical filtering techniques [45,46].

4.2. Key Space

Currently, a key space must contain at least 2^{218} elements [47] to ensure that an encryption algorithm can withstand brute force attacks with a certain level of security. Our approach offers two options: a single key or two independent keys.

4.2.1. Remarks on Pixel Shuffling

The data size to be managed in this work is a matrix of $Z = N \times M$ integer values, which can be within the range [0–255], that is, 256 different values.

Let us assume that each pixel value appears a certain number of times, denoted as R_i; that is, $(R_1, R_2, R_3, ..., R_{256})$, where R_i can take a value within the interval [1–Z], with the restriction that the sum of all values R_i will equal Z.

Let us assume that we start with the Z pixels placed in a bag, and we extract them one by one without replacement to place them into a matrix of size $N \times M$ pixels. Therefore, Z extractions must be performed. The extractions are assumed to be statistically independent.

- Probability that the first selected pixel matches the first pixel of the image: Suppose the first pixel has the value j within the range [0–255], and it appears R_j times; then

$$P_1 = \frac{R_j}{N \times M} \tag{4}$$

- Probability that the second extraction matches the second pixel of the image: Suppose the second pixel has the value k within the range [0–255], and it appears R_k times; then

$$P_2 = \frac{R_k}{(N \times M) - 1} \tag{5}$$

  However, it is also possible that it has the value j, in which case, the probability would be

$$P_2 = \frac{R_j - 1}{(N \times M) - 1} \tag{6}$$

- Following this reasoning, the probability that the L-th extraction matches the L-th pixel of the image: Suppose the L-th pixel has the value m within the range [0–255], and it appears R_m times but has already appeared n times previously; then

$$P_L = \frac{R_m - n}{(N \times M) - (L - 1)} \tag{7}$$

Iterating this process, the final probability of successfully reconstructing the image by extraction would be given by the probability of matching all the pixels of the CGH; that is,

$$P = \prod_{i=1}^{Z} P_i = \frac{\prod_{i=1}^{Z} R_i!}{Z!} \tag{8}$$

The combinatorial explosion depends on the range of values that can be taken and the size of the matrix. In our case, the range is [0–255] and the size is 1920 × 1080. To make an estimation, we know that the histograms of the phases encoded in a png file are practically uniform; that is, the frequency values in these histograms tend to satisfy the following equality: $R_1 = R_2 = ... = R_{256} = Z/256$. In this case, we can make the following estimation:

$$P = \frac{(Z/256)!^{256}}{Z!} = \frac{8100!^{256}}{2073600!} \tag{9}$$

The factorial is an extraordinarily large number. Using Stirling's approximation [48], an estimation can be made as follows:

$$n! \approx \sqrt{2\pi n}(\frac{n}{e})^n \tag{10}$$

To determine the size of this number, we can calculate the number of digits it has.

$$\log_{10}(n!) \approx n \log_{10}(n) - n \tag{11}$$

A quantitative approximation for the images we are using would yield the following result:

$$P \approx \frac{(10^{23558})^{256}}{10^{11024760}} = \frac{10^{6030848}}{10^{11024760}} = \frac{1}{10^{4993912}} \tag{12}$$

The denominator contains nearly 5 million digits, making the probability of reconstructing the $N \times M$ matrix practically zero. If a processing machine required 10^{-12} s per possibility, it would take $10^{4993912} \times 10^{-12}$ s, an extraordinarily long period compared with any human or even geological time scale (for reference, the age of the universe is in the order of 10^{17} s).

This pixel shuffling method of a scene, in which a mathematical transformation operation has been previously applied (the synthesis of a CGH can be likened to this operator), already provides a fairly robust encryption against statistical analysis (the phase distribution in a histogram is practically flat) or brute force attacks.

4.2.2. Key Size Issues

The proposed key is a vector of $N \times M$ integer values whose size increases as a function of the CGH resolution, which may result in keys of considerable size. One should consider the option of compressing this information using arithmetic encoding techniques [49] that guarantee to avoid loss of information in order to have a shorter key. Other options to consider involve the use of chaotic functions [18,21] to rearrange the pixels of the CGH.

4.2.3. Remarks on DRPE Phase Mask

Let us recall that we have the key, which is a vector containing the permutations of the pixels, and a vector in which the permuted phases are stored. Now we will modify these values as follows: Another vector will be constructed to store the modifications of the phase values, based on the vector containing the pixel permutations. Since Z is a number greater than the phase range encoded in the CGH [0–255], the proposed procedure is to truncate (modulo 256) each component of the pixel shuffling vector (the key) and add it to the phase in the same position in the phase vector. In this way, for each pixel in the CGH, a new phase value is obtained, which can be within the range [0–255]. A brute force attack would involve modifying the positions and values of the pixels.

Following a similar approach to the previous subsection, we assume that we have Z elements of a matrix with values in the range [0–255]. Adding a random phase implies that each of these Z elements can have any value within the mentioned range. The probability of recovering the original phase of pixel i in the CGH is

$$P_{phase_i} = \frac{1}{256} \tag{13}$$

Since each pixel is an independent case, the probability of correctly recovering all the phases would be

$$P_{phase_{SLM}} = \prod_{i=1}^{Z} P_{phase_i} = \frac{1}{256^{Z}} = \frac{1}{256^{(1920\times1080)}} \tag{14}$$

The probability of finding the correct phase match is also very small (the denominator has nearly 5 million digits) and adds another layer of security to the CGH encryption, breaking any pattern that could have existed in the phase histogram.

4.3. Key Sensitivity

Key sensitivity is a crucial property in image encryption algorithms, as it quantifies how small variations in the key can lead to significantly different results. This study has not focused on this characteristic, as phase masks or pixel shuffling vectors can be generated using algorithms described in the literature that inherently possess this property [18,21].

4.4. Sensitivity vs. Robustness of Information

The very nature of CGHs makes them highly robust to attacks that aim to partially modify the stored information. In other words, small variations in the 3D scene (understood as plaintext) do not significantly disturb the result stored in the CGH. Likewise, small modifications in the CGH do not lead to the loss of substantial information.

Processes that partially modify the number of shuffled pixels in which their phases have been altered can provide insights into how the CGH behaves under progressively larger disturbances.

A method to quantify this statement is by using the Pearson Correlation Coefficient to measure the similarity between images. In Figure 7a, this coefficient is shown as a function of the percentage of shuffled pixels, pixels with modified phase, and pixels where both phase and position are modified, between the original reconstructed image (Figure 5a) and those obtained with different percentages. Only when all pixels are shuffled is the absence of correlation guaranteed. For the calculation of the Correlation Coefficient (CC) shown in Figure 7a, the reference image is Figure 5a. In the case of 0% shuffled pixels, it is compared with itself ($CC = 1$). This value decreases, reaching ($CC = 0$) when 100% of the pixels are modified (Figure 7a).

Since all points in the scene contribute to all pixels in the CGH, a nearly complete shuffle is necessary to completely hide the recorded information. The same is true for an encryption process where only the phase mask is used to hide the information.

4.5. Pixel Correlation

One of the characteristics of conventional images is that they exhibit high correlation between adjacent pixels. In CGHs, this correlation is very small or practically absent, since, currently, the pixel size is much larger than the wavelength used to calculate the stored phases (8 µm vs. 0.633 µm). These phases are calculated with the contribution of millions of points, resulting in completely different values between pixels (see Figure 2b).

4.6. Pixel Distribution

Another property of the phase information stored in holograms is that it has a pixel distribution in which it is difficult to identify structures, correlations, or patterns (much more so than in images obtained directly from scenes). The reason, once again, is that the sum of multiple contributions from the scene to each pixel makes it virtually impossible to predict the value of adjacent pixels based on the value of any specific pixel. Figure 7b shows the histogram of the encrypted CGH, where it is observed that the phase value distribution is nearly flat, which helps to hinder statistical analysis-based hacking attempts.

4.6.1. Information Entropy

The entropy of a grayscale image p for all elements of an image x_i can be calculated as

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i) \tag{15}$$

The entropy values obtained for the images involved in this work can be found in Table 1. These data confirm that the entropy of the information stored in the CGH is higher than in the images and does not change during the encryption process. The difference in entropy between the original image and the recovered images is due to the limitations imposed by diffraction (speckle, pixel sizes, etc.).

Table 1. Entropy calculated for the images shown in this work (using the "entropy" function in MATLAB).

Description	Figure	Entropy H(X)
Image from 3D scene [1]	Figure 2a	7.067178
CGH	Figure 2b	7.994257
Encrypted CGH	Figure 4a	7.994281
Image from recovered scene (simulation)	Figure 5a	5.389448
Image from recovered scene (lab) [1]	Figure 5b	6.072168

[1] Using only grayscale values.

The entropy values between the CGH and the encrypted CGH are virtually the same because integer values are used to shuffle or apply phase masks to the PNG file of the CGH, preserving the same information throughout the encryption and decryption process.

4.6.2. Efficiency Analysis

The complete generation process shown in Figure 1 is computationally demanding, with the CGH calculation being the most time-consuming part. The specifications of our computer are as follows: CPU Intel i9-9900K (Intel Corp., Portland, OR, USA) (16 cores) GPU NVIDIA GeForce RTX 2060 (NVIDIA, Taipei, Taiwan.) (1920 CUDA cores), and 32 GB of RAM. The characteristics of the scene in Figure 2a are as follows: 116,090 triangles, 4 light sources, and 1920 × 1080 px. For this scene, the computation times (using CUDA for amplitude and phase calculation) are as follows: For 240,000 points, 295 s with double precision and 101 s with single precision. For 1.1 M points, 1600 s and 600 s, respectively. The calculations were performed using a code implemented in C++23. Table 2 summarizes the computation times for each part of the process described in this work.

For the encryption process (to add random phase and pixel shuffling), the same computer and Matlab R2024b were used. The encryption and decryption times do not exceed 1 s in any case, so the bottleneck in performance lies in the generation of the CGH.

Table 2. Computation times in seconds for the steps of Figure 1 using 10 samples per pixel for the generation of the scene. Data for 1920 × 1080 pixels for both the scene and the CGH size.

Process	t(s.)
(a) Scene synthesis	26.55
(b) CGH synthesis	1600.37 [1]
(c) Encryption	0.32
(d) Decryption	0.32
(e) Scene simulation	0.78

[1] Using double precision for coding amplitude and phase.

Since the computation of the CGH is a linear process, the computational cost is proportional to the number of pixels used for its synthesis. Table 3 presents the computation times that confirm this behavior. It is important to emphasize that these times are indicative and depend on both the hardware used and the programming languages employed to implement each part of the process.

Table 3. CGH computation times in seconds vs. CGH resolution using double precision for coding amplitude and phase.

CGH Size (in Pixels)	t(s.)
1920 × 1080	1600.37
1024 × 1024	809.08
512 × 512	204.27
256 × 256	56.06
64 × 64	16.27

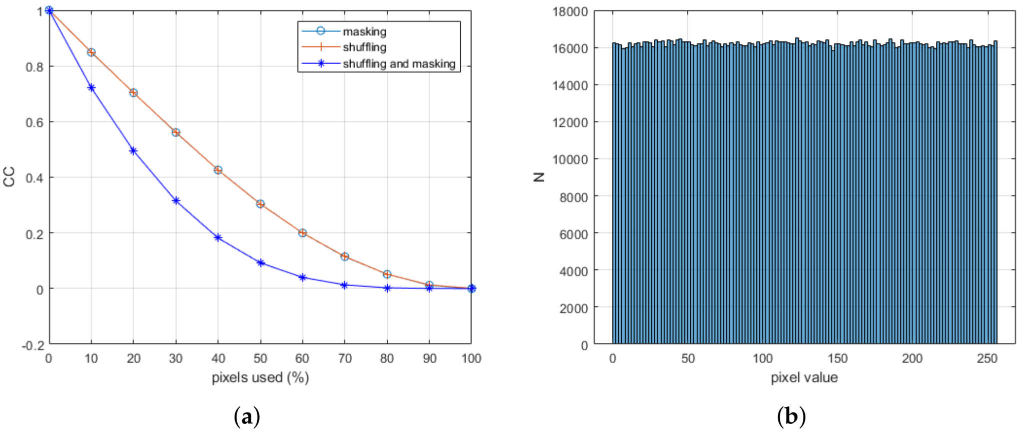


Figure 7. (**a**): Pearson Correlation Coefficient (PCC) as a function of the percentage of shuffled pixels (orange line) or when only the phases are modified (light blue). The behavior of both graphs is identical. PCC when both are used (dark blue line). (**b**): Distribution of grayscale levels (phases) of the encrypted CGH in Figure 4a.

These graphs can aid in managing the trade-off between encryption computation time and the required level of security. Additionally, other factors complicate the process of unauthorized decryption, such as the wavelength used for CGH synthesis or the geometry of the recording process (the distance between the scene and the CGH plane). In a trial-and-error process, the computation time should also account for the cost of calculating the convolution integral to reconstruct the scene from CGH.

5. Conclusions

Encoding information using CGH and random phases (DRPE) has proven to be a robust technique for information encryption in the past. However, the time required to find the decryption key with brute force attacks decreases as the available computing power increases.

We present an evolution of DRPE that adds the random permutation of pixels. This proposal allows the use of a single key to generate the CGH that serves as ciphertext or the use of two independent keys: one for the permutation and another for the synthesis of the random phase mask.

The entire process can be simulated on a computer or demonstrated in a laboratory. The permutation possibilities add a new layer of complexity against attacks.

We have analyzed the performance of this proposal through a visual assessment. We have significantly increased the size of the key space compared with the DRPE proposals, while preserving the robustness of the information by storing it using CGH. In this context, variables such as pixel correlation and image entropy show substantial improvement in the field of encryption.

From a weakness perspective, our proposal offers a key size larger than those in previous approaches; however, it remains compatible with using some of these proposals for key generation. This presents an area for future work, particularly with encryption algorithms based on chaotic systems. Additionally, another potential direction involves applying this process to color CGH, which would essentially triple the information size.

Encoding the information in CGH makes the generated ciphertext robust to localized modifications, the random phase modification makes the phase histogram distribution more resilient to statistical attacks, and the shuffling of already-encrypted pixels significantly increases the combinatorial possibilities to explore, rendering a brute force attack unfeasible.

Author Contributions: Conceptualization, A.B. and F.J.S.; methodology, A.B.; software, A.B.; validation, A.B. and F.J.S.; formal analysis, A.B. and F.J.S.; investigation, A.B. and F.J.S.; resources, A.B. and F.J.S.; data curation, A.B. and F.J.S.; writing—original draft preparation, A.B.; writing—review and editing, F.J.S.; visualization, A.B. and F.J.S.; supervision, A.B. and F.J.S.; project administration, A.B.; funding acquisition, A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This initiative is carried out within the framework of the EU-funded "Plan de de Recuperación, Transformación y Resiliencia" (Next Generation), Instituto Nacional de Ciberseguridad (INCIBE) (Spain).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Acknowledgments: The authors would like to thank Diego Sanz (trainee student of the Department of Computer Science at the University of Zaragoza) for his contribution in the design and synthesis of Figure 2a.

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

References

1. Goodman, J.W. *Introduction to Fourier Optics*, 3rd ed.; Roberts & Co. Publishers: Englewood, CO, USA, 2017; Volume 1.
2. Singh, M.; Singh, A.K. A comprehensive survey on encryption techniques for digital images. *Multimed. Tools Appl.* **2023**, *82*, 11155–11187. [CrossRef]
3. SaberiKamarposhti, M.; Ghorbani, A.; Yadollahi, M. A comprehensive survey on image encryption: Taxonomy, challenges, and future directions. *Chaos Solitons Fractals* **2024**, *178*, 114361. [CrossRef]
4. Javidi, B.; Carnicer, A.; Yamaguchi, M.; Nomura, T.; Pérez-Cabré, E.; Millán, M.S.; Nishchal, N.K.; Torroba, R.; Barrera, J.F.; He, W.; et al. Roadmap on optical security. *J. Opt.* **2016**, *18*, 083001. [CrossRef]
5. Wang, W.; Wang, X.; Xu, B.; Chen, J. Optical image encryption and authentication using phase-only computer-generated hologram. *Opt. Lasers Eng.* **2021**, *146*, 106722. [CrossRef]
6. Daemen, J.; Rijmen, V. *The Design of Rijndael*; Springer: Berlin/Heidelberg, Germany, 2002. [CrossRef]
7. Erkan, U.; Toktas, A.; Memiş, S.; Lai, Q.; Hu, G. An image encryption method based on multi-space confusion using hyperchaotic 2D Vincent map derived from optimization benchmark function. *Nonlinear Dyn.* **2023**, *111*, 20377–20405. [CrossRef]
8. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
9. Paul, G.; Maitra, S. *RC4 Stream Cipher and Its Variants*; CRC Press: Boca Raton, FL, USA, 2011.
10. Brakerski, Z.; Gentry, C.; Vaikuntanathan, V. Fully Homomorphic Encryption without Bootstrapping. *ACM Trans. Comput. Theory* **2014**, *6*, 1–36. [CrossRef]

11. Research, M. Microsoft SEAL (Simple Encrypted Arithmetic Library). 2020. Available online: https://github.com/microsoft/SEAL (accessed on 10 February 2025).
12. Hankerson, D.; Menezes, A.J.; Vanstone, S. *Guide to Elliptic Curve Cryptography*; Springer: Berlin/Heidelberg, Germany, 2006.
13. Bernstein, D.J.; Lange, T.; Peters, C. Post-Quantum Cryptography. *Nature* **2017**, *549*, 188–195. [CrossRef]
14. Bos, J.W.; Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schanck, J.M.; Schwabe, P.; Seiler, G.; Stehlé, D. CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM. In Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), London, UK, 24–26 April 2018; pp. 353–367. [CrossRef]
15. Hoffstein, J.; Pipher, J.; Silverman, J.H. NTRU: A Ring-Based Public Key Cryptosystem. In *Algorithmic Number Theory*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 267–288. [CrossRef]
16. Lyubashevsky, V.; Lepoint, T.; Seiler, G. CRYSTALS-Dilithium: Digital Signatures from Module Lattices. In Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), London, UK, 24–26 April 2018; pp. 356–373. [CrossRef]
17. Zhang, L.; Liao, X.; Wang, X. An image encryption approach based on chaotic maps. *Chaos Solitons Fractals* **2005**, *24*, 759–765. [CrossRef]
18. Feng, W.; Yang, J.; Zhao, X.; Qin, Z.; Zhang, J.; Zhu, Z.; Wen, H.; Qian, K. A Novel Multi-Channel Image Encryption Algorithm Leveraging Pixel Reorganization and Hyperchaotic Maps. *Mathematics* **2024**, *12*, 3917. [CrossRef]
19. Feng, W.; Zhang, J.; Chen, Y.; Qin, Z.; Zhang, Y.; Ahmad, M.; Woźniak, M. Exploiting robust quadratic polynomial hyperchaotic map and pixel fusion strategy for efficient image encryption. *Expert Syst. Appl.* **2024**, *246*, 123190. [CrossRef]
20. Qian, K.; Xiao, Y.; Wei, Y.; Liu, D.; Wang, Q.; Feng, W. A Robust Memristor-Enhanced Polynomial Hyper-Chaotic Map and Its Multi-Channel Image Encryption Application. *Micromachines* **2023**, *14*, 2090. [CrossRef]
21. Feng, W.; Wang, Q.; Liu, H.; Ren, Y.; Zhang, J.; Zhang, S.; Qian, K.; Wen, H. Exploiting Newly Designed Fractional-Order 3D Lorenz Chaotic System and 2D Discrete Polynomial Hyper-Chaotic Map for High-Performance Multi-Image Encryption. *Fractal Fract.* **2023**, *7*, 887. [CrossRef]
22. Lo, H.K.; Curty, M.; Tamaki, K. Secure Quantum Key Distribution. *Nat. Photonics* **2014**, *8*, 595–604. [CrossRef]
23. Hao, W.; Zhang, T.; Chen, X.; Zhou, X. A hybrid NEQR image encryption cryptosystem using two-dimensional quantum walks and quantum coding. *Signal Process.* **2023**, *205*, 108890. [CrossRef]
24. Wang, X.; Zhang Y.; Bao X. A novel chaotic image encryption scheme using DNA sequence operations. *Opt. Lasers Eng.* **2015**, *73*, 53–61. [CrossRef]
25. Meng, F.; Gu, Z. A Color Image-Encryption Algorithm Using Extended DNA Coding and Zig-Zag Transform Based on a Fractional-Order Laser System. *Fractal Fract.* **2023**, *7*, 795. [CrossRef]
26. Cheng, L.; Yu, W.; Zhang, Z.; Zhang, Y. Image encryption using deep learning. *Neurocomputing* **2019**, *338*, 144–152. [CrossRef]
27. Wang, C.; Zhang, Y. A novel image encryption algorithm with deep neural network. *Signal Process.* **2022**, *196*, 108536. [CrossRef]
28. Zhou, Q.; Wang, X.; Jin, M.; Zhang, L.; Xu, B. Optical image encryption based on two-channel detection and deep learning. *Opt. Lasers Eng.* **2023**, *162*, 107415. [CrossRef]
29. Chen, W.; Javidi, B.; Chen, X. Advances in optical security systems. *Adv. Opt. Photon.* **2014**, *6*, 120–155. [CrossRef]
30. Nishchal, N.K. *Optical Cryptosystems*; IOP Publishing Ltd.: Bristol, UK, 2019; pp. 1–180. [CrossRef]
31. Singh, H.; Yadav, A.K.; Vashisth, S.; Singh, K. Double phase-image encryption using gyrator transforms, and structured phase mask in the frequency plane. *Opt. Lasers Eng.* **2015**, *67*, 145–156. [CrossRef]
32. Refregier, P.; Javidi, B. Optical image encryption based on input plane and Fourier plane random encoding. *Opt. Lett.* **1995**, *20*, 767–769. [CrossRef]
33. Tajahuerce, E.; Javidi, B. Encrypting three-dimensional information with digital holography. *Appl. Opt.* **2000**, *39*, 6595–6601. [CrossRef]
34. Takeda, M.; Nakano, K.; Suzuki, H.; Yamaguchi, M. Encrypted imaging based on algebraic implementation of double random phase encoding. *Appl. Opt.* **2014**, *53*, 2956–2963. [CrossRef]
35. Carnicer, A.; Montes-Usategui, M.; Arcos, S.; Juvells, I. Vulnerability to chosen-cyphertext attacks of optical encryption schemes based on double random phase keys. *Opt. Lett.* **2005**, *30*, 1644–1646. [CrossRef]
36. Hamadi, I.A.; Jamal, R.K.; Mousa, S.K. Image encryption based on computer generated hologram and Rossler chaotic system. *Opt. Quantum Electron.* **2022**, *54*, 33. [CrossRef]
37. Hu, T.; Ying, Y.; Sun, X.; Jin, W. The recovery scheme of computer-generated holography encryption–hiding images based on deep learning. *Opt. Commun.* **2023**, *529*, 129100. [CrossRef]
38. Nomura, T.; Javidi, B. Optical encryption based on the input phase mask designed for the space bandwidth of the optical system. In *Optical Information Systems III*; Javidi, B., Psaltis, D., Eds.; International Society for Optics and Photonics, SPIE: Bellingham, WA, USA, 2005; Volume 5908, p. 59080B. [CrossRef]
39. Kajiya, J.T. The rendering equation. *SIGGRAPH Comput. Graph.* **1986**, *20*, 143–150. [CrossRef]
40. PLUTO-2 Phase Only Spatial Light Modulator (Reflective) | HOLOEYE Photonics AG. Available online: https://holoeye.com/products/spatial-light-modulators/pluto-2-1-lcos-phase-only-refl/ (accessed on 10 February 2025).

41. Magallón, J.A.; Blesa, A.; Serón, F.J. Monte–Carlo Techniques Applied to CGH Generation Processes and Their Impact on the Image Quality Obtained. *Eng. Rep.* **2025**, *7*, e13109. [CrossRef]
42. Durstenfeld, R. Algorithm 235: Random permutation. *Commun. ACM* **1964**, *7*, 420. [CrossRef]
43. Hua, Z.; Zhou, Y.; Huang, H. Cosine-transform-based chaotic system for image encryption. *Inf. Sci.* **2019**, *480*, 403–419. [CrossRef]
44. Pi, D.; Liu, J.; Wang, Y. Review of computer-generated hologram algorithms for color dynamic holographic three-dimensional display. *Light. Sci. Appl.* **2022**, *11*, 231. [CrossRef] [PubMed]
45. Bianco, V.; Memmolo, P.; Leo, M.; Montresor, S.; Distante, C.; Paturzo, M.; Picart, P.; Javidi, B.; Ferraro, P. Strategies for reducing speckle noise in digital holography. *Light. Sci. Appl.* **2018**, *7*, 48. [CrossRef]
46. Blinder, D.; Birnbaum, T.; Ito, T.; Shimobaba, T. The state-of-the-art in computer generated holography for 3D display. *Light. Adv. Manuf.* **2022**, *3*, 572–600. [CrossRef]
47. Alvarez, G.; Li, S. Some basic cryptographic requirements for Chaos-Based Cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
48. Knuth, D.E. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*, 3rd ed.; Includes discussion on Stirling's approximation in combinatorial analysis; Addison-Wesley: Boston, MA, USA, 1997.
49. Witten, I.H.; Neal, R.M.; Cleary, J.G. Arithmetic coding for data compression. *Commun. ACM* **1987**, *30*, 520–540. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.