International Journal of Basic and Applied Sciences, 14 (3) (2025) 178-184

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/t9dxpr23
Research paper

A Novel Approach of Cryptography Via Z-Buffer Method

Deepika Gautam, Vipin Saxena *

Department of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, Uttar Pradesh, India
**Corresponding author E-mail: profvipinsaxena@gmail.com*

Received: May 28, 2025, Accepted: July 9, 2025, Published: July 20, 2025

Abstract

As the reliability on the cloud grows in modern technology, the importance of robust security for data to reduce the risk of breaching and unauthorized access has also become more essential. A novel approach is presented over the cloud to secure textual data over the cloud ecosystem by adapting the Z-Buffer algorithm, which is commonly used in computer graphics. The proposed technique is incorporating complex number-based key generation and hyperbolic encryption and decryption. The system shall be effective and scalable for data organization, which is developed through advanced concepts of mathematics. The performance of the proposed method is analyzed using primary metrics like central processing unit utilization and latency, as it demonstrates the ability to enhance security and proper utilization of resources across different file sizes, supporting real-world cloud applications and providing practical solutions to sensitive data. The pro-posed work is apart from previous work because of the fusion of graphic algorithm with security method, showing a new path for future possibility in cloud data security. The research work addresses the present challenges in security and also lays down the foundation for future breakthroughs in the area of security. By filling the gap between security and computer graphics, the presented approach offers a new perspective to secure data over cloud-dependent environments.

Keywords: *Cloud Storage; Cryptography; Data Security; Hyperbolic Equation; Resource Optimization; Z-Buffer.*

1. Introduction

The growing need for data security in cloud storage is underscored by the rapid increase in data volume and common cyber threats targeting the sensitive information. As organizations rely on cloud computing for scalable and cost-effective data management, significant security challenges are also faced, including unauthorized access and data breaches from both attackers and intruders. Advanced security measures, such as encryption techniques, biometric authentication, and access control mechanisms, are essential to protect data integrity and confi-dentiality. Moreover, the integration of artificial intelligence (AI) into cloud security architectures is emerging as a promising approach to enhance data protection against evolving threats. The implementation of robust security models, including hyperchaotic encryption and hashing methods, further mitigates risks associated with data outsourcing.

Current cryptographic methods face significant limitations in handling complex data types in cloud environments. Traditional encryption techniques, including AES and RSA, while effective for basic data security, struggle with efficiency and scalability when applied to large datasets. The challenge of efficiently processing encrypted data without compromising security remains unresolved, so several innovative approaches that combine software and hardware solutions are developing continuously.

The Z-buffer algorithm is a widely used technique in computer graphics for managing image depth information, ensuring that the closest objects are rendered in front of those further away. Recent advancements have introduced the Z-buffer, which enhances ray tracing perfor-mance by accommodating complex scenes with features like transparency and shadows. The integration of complex number-based encryp-tion methods offers a novel approach to securing data. The proposed method utilizes Z-transformations for encryption, ensuring both security and speed in data transmission. Layers of complexity and randomness are also introduced, making it difficult to break security, which enhances data security and data efficiency and reduces redundancy. The method is beneficial for several real-world applications, such as:

a) **Cloud Data Security:** Secure data storage and transmission over cloud servers.
b) **IoT security:** provides better security to connected devices and edge computing.
c) **Media security:** is the security over sensitive textual data such as financial data etc.
d) **Decentralized storage and blockchain:** contain cryptographic frameworks which must enhance the blockchain and data verification, validation, and consistency.

A crucial computer graphics algorithm is Z-buffer, also named depth buffer, because while rendering scenes, it gives depth information, ensuring objects are properly visible. The proposed method maintains a depth value for every pixel of the 3D object on the view plane. Z-buffers are widely used by real-time graphics applications like virtual reality and video games because they are simple and effective (1). While rendering, the depth of incoming pixels and the Z-buffer were compared, and the value is stored. If any new pixel is closer, then it updates the value inside the Z-buffer and the respective color buffer (2), and caching techniques are enhanced, such as Z-cache. Z-cache

helps to speed up the depth values while improving the overall performance. Some advanced techniques like Z-culling are used to optimize the performance by decreasing the use of memory bandwidth with the help of efficient depth value (3). Several applications adapted Z-buffer techniques like computer-generated holography, in which path length rays were managed perfectly during reconstruction of 3D images (4). In addition, hardware progress, mainly in mobile having graphics processing unit (GPU) platforms, has increased the performance of z-buffer usage, allowing advanced graphic techniques in real time (5). A convolutional neural network model in Z-buffer detects and categorizes important parts in SAR images to improve accuracy in comparison to existing methods (6). Binary space partition tree is also a method for hidden space detection and recursion in 3D images like Z-buffer (7).

Complex numbers play a vital role in numerous fields, such as mathematics and computer science. In math, complex numbers helped broaden the real number system by figuring out the solution of equations that lack real roots, such as $x^2 + 1 = 0$. Complex analysis (8) deals with functions of complex variables and was pivotal in various areas, including engineering and physics. The authors detailed properties such as analyticity and contour integration, which were essential for understanding complex functions. Complex exponential (9) functions facilitate the representation and manipulation of signals in the frequency domain. A simplified equation (10) demonstrated that transformations in 2D or 3D graphics can be elegantly handled using complex numbers, which emphasizes that the use of complex numbers is fundamental to quantum mechanics (11), particularly in the formulation of quantum states and operations. Complex-Valued Neural Networks (CVNN) (12) can improve the performance of learning algorithms, applications, and activation functions and leverage the properties of complex numbers to enhance feature representation, particularly in tasks involving oscillatory data.

Cryptography (13) has evolved significantly in recent years, adapting to the increasing complexity of cyber threats and the demands of modern communication. A significant trend was the rise of Elliptic Curve Cryptography (ECC) homomorphic encryption, which allowed computations to be performed on encrypted data without decrypting it first (14) and more recent works, such as making strides in optimizing these systems for practical use, facilitating secure data processing in cloud computing environments (15). Blockchain technology continues to influence cryptography as well, with advancements in decentralized identity and zero-knowledge proofs. Zcash, Loopring 3.0, and Filecoin (16) exemplify the application of zero-knowledge proofs, enabling users to prove possession of information without revealing the information itself, thus enhancing privacy. Furthermore, the integration of machine learning into cryptographic practices was an innovative idea. Recent studies (17) have explored the use of machine learning for anomaly detection in cryptographic protocols, enhancing security through predictive analytics. Finally, the adoption of quantum key distribution (QKD) is advancing, with protocols such as BB84 being refined to ensure secure key exchange over potentially compromised channels. Bennett and Brassard (18) laid the groundwork, and contemporary research continues to improve QKD systems efficiency and practicality. Several other hybrid algorithms were also presented, which helped in cryptography, such as the combination of symmetric and asymmetric algorithms with genetic algorithms (19), hash and ElGamal with fingerprint authentication (20), DNA and the Paillier method (21), and RSA and AES (22). Saving space on the server to optimize storage space by removing duplicate files (23) and binary search method to compress the file with quick retrieval and without data loss (24) can help in securing files more efficiently using the fuzzy ElGamal method (25). Several hybrid cryptographic method, combining AES, ECC, and SHA-256, enhances text encryption efficiency but requires further optimization for image encryption performance [26]. 47 % of present research were related to elliptic-curves, and an increase of inquiries of hybrid encryption systems-especially image protection-was observed since last year (27). The methods like Elliptic Curve Cryptography (ECC) and homomorphic encryption present notable limitations in the context of large-scale cloud-based applications (28). ECC, although efficient in reducing key size, often exhibits increased processing overhead when applied to multimedia data, limiting its suitability for high-throughput cloud environments (29). Similarly, while homomorphic encryption enables computation on encrypted data, it remains computationally intensive, which impacts its practicality for real-time applications in the cloud (30). Recent advancements in post-quantum encryption, including lattice-based methods, offer resistance against quantum attacks but face integration challenges due to large key sizes and system overhead (31). In contrast, the proposed Z-buffer-based encryption method utilizes layered segmentation and parallelism, ensuring computational efficiency and scalability. Unlike AES, which encounters difficulties in handling diverse and large multimedia datasets (32), the Z-buffer method maintains performance by leveraging depth-buffering strategies for structured encryption and decryption operations across varying file types. The importance of a thorough interaction with cryptanalysis, multi-party computation and side-channel attacks, thus indicating the relevant community is continuing to transform cryptographical research (33). The further analysis of modern cryptographic algorithms highlights the fact that they still offer difficulties simultaneously on the level of security and performance and proves once again the necessity of the same conclusion that the work of researchers to fix the drawbacks of the algorithms and make them more resilient as vital (34). In the field of money and technology, the current attitude of the practitioners and regulators is the belief that quantum-resistant algorithms, homomorphic encryption, and artificial-intelligence capacities will transform the industry and all these take place along with the parallel development of blockchain and biometric security, all within a legal and ethical paradigm that is continually changing (35).

2. Methodology

The Z-buffer algorithm is majorly used for objects in 3D scenes for better visibility in computer graphics. It decides which object to display and which to hide. The Z-buffer stores the z-value of each pixel of the screen. The Z-value is the distance between the camera and the object presented, shown in Fig. 1. While rendering many objects, the Z-value of every pixel is compared with the stored Z-buffer value. If the value of the new object is smaller than the stored Z-buffer, then it will update the new, smaller value. The process is repeated until each pixel is updated and the final image is correctly represented. Following are the steps of the Z-buffer algorithm.

Algorithm 1:Z_buffer()

```
Initialization of depth and color values
Put the pixel's depth value as d(m, n)= ∞
Put the pixel's color value as c(m, n)= backdrop color
Process of every polygon
        For every polygon
        Pixel inside the polygon's projection:
                Depth(z) of polygon (x, y) correlated to pixel (m, n)
        If depth valve (z) < pixel's depth value d(m, n)
         |Set the depth value as d(m, n)=z
        |Set the color value as c(m, n)=color
```

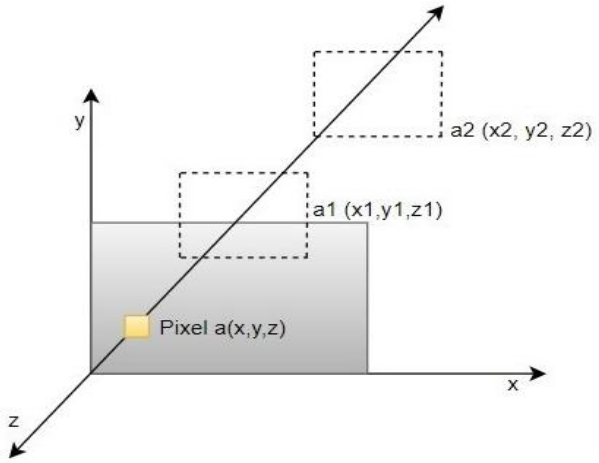

Fig. 1: Z-Buffer Depth Handling for Pixel Rendering.

The proposed method used the core principle of the above algorithm to enhance data security. By applying the concept of three different axes and using the value of them, they created a robust encryption method to ensure secure data transmission. The method is an enhanced layered encryption process of security protocol using the Z-buffer concept originally utilized to handle depth in 3D rendering. Critical data is encrypted after dividing it into coordinates. The Z-buffer is created with a 2D array, and to represent a 3D object, x, y, and z coordinates are used. During 3D object exposure. The Z-buffer helps to determine which object is in front or behind by storing the depth of each pixel.
X: object placed on horizontal axis
Y: object place on vertical axis
Z: Depth or distance from viewer
As a Z-buffer handles layers of objects in a 3D environment, similarly a security method is developed, which is depth-based encryption, in which certain layers or coordinates are secured based on their importance and access level. According to the proposed method, the new meaning of coordinates is.
X:indexing
Y: frequency
Z: storing ASCII value
The method is used to encrypt and decrypt text files. The method is a combination of complex number key generation, hyperbolic function for encryption and decryption, and ASCII-based Z-buffer, shown in Fig.2 and Fig.3 .To improve security, a complex number mechanism is used on data. The key generation steps are mentioned below.

Key Generation

1) The matrix of the z-buffer is converted to a pair of index and ASCII values of characters.
2) Each pair is represented as a complex number.
3) Complex numbers are further categorized into positive (pos_CN) and negative (neg_CN).
4) A binary tree is formed recursively using positive numbers as right-side nodes and negative numbers as left-side nodes.
5) At last encryption key is extracted from the left-side node of the merged binary tree.

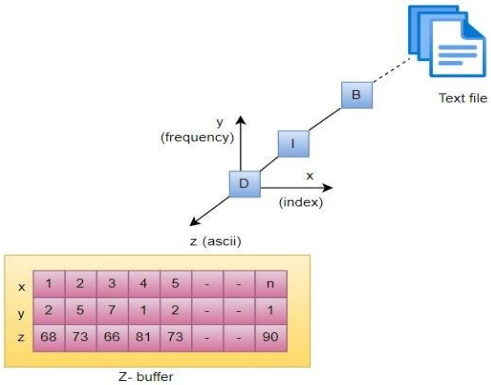

Fig. 2: Z-Buffer Construction for Text Files Encryption.

Fig. 3: Z-Buffer-Based Encryption and Decryption Process.

The encryption and decryption are obtained by mathematical functions of hyperbolics to alter a character's ASCII value, providing secure transition of data.

Encryption Process

1) A scaling factor is introduced to normalize data.
2) Scaled ASCII values are computed.
3) From scaled values, scaled keys are generated.

4) Hyperbolic encryption is applied in scaled keys, and value is stored in the output buffer.

Decryption process

1) Encrypted data and complex keys are being restored.
2) Inverse hyperbolic is used to obtain original data.
3) Scaling operation is reversed to get ASCII value again.
4) The ASCII value is stored in the buffer in the same place from where it were taken to reconstruct the same data.

The method ensures that minor changes can affect the significant variation while decrypting data, which strengthens security against brute-force attacks. The proposed combination of security gave a multi-layered security model that helps:

1) In efficient computation while protecting big files.
2) Including randomness in complex numbers to make strong and unpredictable keys.
3) Reduction of encryption and decryption latency for real-world application.

```
Algorithm 2:Z_buffer_security()
Initialize a list z_buff = [].
Input the data and calculate frequency of every character.
For each i, char in the data ( i = index of character and char is the character):
Set x = i (index of the character).
Set y = freq [char] (the frequency of the character).
Set z = ASCII [char] (the ASCII value of the character).
z_buff.append [x, y, z].
Store z_buff in 2D-matrix form.
Separate the array into pairs to generate complex numbers.
CN.append(comp(arr[i], arr[i + 1])
Construct binary trees with recursive function.
If CN.real>= 0, append to pos_CN.
else append to neg_CN.
Initialize merge_tree with small_pos_CN as the root.
Set merge_tree.left = neg_tree and merge_tree.right = pos_tree.
Generate the complex key by rounding the real and imaginary parts of the left node's value in the combined tree.
return comp(a, b).
Initialize scale_fac = 0.001.
enc_asc(asc_val, key)
scaled_val = scale_fac * asc_val
scaled_key = scale_fac* abs(key)
enc_val= cosh(scaled_key * scaled_val)
enc_data(data, key)
appendenc_asc (byte, key) to enc_asc_val.
return enc_asc_val
dec_asc(enc_val, key)
scaled_key = scale_fac* abs(key)
dec_val = arcosh (enc_val) / scaled_key.
dec_real = round(dec_val.real / scale_fac)
dec_data(enc_val, key)
Initialize dec_data = [].
append dec_asc(enc_val, key) to dec_data.
return dec_asc(dec_data).
End Procedure
```

The above algorithms stated the process of generation of keys by dividing the array into two pairs of complex numbers. Arrays are divided on the basis of positive and negative real parts of complex numbers, and further, a binary tree is built in a recursive manner. Using the array of positive complex numbers, build a tree with the negative as a left child and the positive tree sits right child. Now generate the complex number with the help of rounding the real and imaginary numbers of the left node from the merged tree. The next step was to start with a scaling factor, and further encryption and decryption functions were defined using hyperbolic functions. Encryption is performed by encrypting ASCII values, and decryption by reversing the process while ensuring that all the elements fell into place during decryption.

3. Results and discussions

The proposed method uses moderate resources to recreate the real-world scenarios. The proposed Z-buffer method is implemented on different sizes of text files to evaluate performance under different data conditions. Table 1 shows the outcomes of different sizes of text files' performance based on encryption/decryption time, CPU utilization, and latency. When the size of files increases, computation time also increases gradually due to increased processing requirements. Further, for larger key sizes varying from 128 to 1024 bits, CPU utilization rises constantly without compromising performance. The performance highlights the efficiency and scalability of the method, showing it maintains computation time, controls CPU utilization, and provides less latency, which makes securing data of various file sizes with optimized resource consumption.

Table 1: Performance Analysis Z-Buffer on Different Text File Sizes

File Size	Key Size (bits)	Encryption Time (sec)	Decryption Time (sec)	CPU Utilization (%)	Latency (sec)
16 KB	128	0.030	0.042	9	0.009
32 KB	128	0.037	0.049	10	0.010
64 KB	128	0.045	0.057	11	0.011
128 KB	128	0.062	0.075	13	0.013
256 KB	256	0.080	0.097	16	0.016
512 KB	256	0.095	0.112	20	0.019
1 MB	256	0.110	0.130	23	0.022

2 MB	256	0.138	0.152	26	0.026
4 MB	256	0.178	0.194	29	0.030
8 MB	512	0.220	0.235	33	0.036
16 MB	512	0.260	0.278	37	0.042
32 MB	512	0.319	0.336	41	0.049
64 MB	512	0.390	0.410	45	0.057
128 MB	1024	0.478	0.499	49	0.068
228 MB	1024	0.565	0.591	53	0.076
256 MB	1024	0.592	0.621	55	0.079
512 MB	1024	0.735	0.762	59	0.090
1 GB	1024	0.880	0.902	62	0.101

Table 2: Performance Analysis of the Hybrid Method and Proposed Method on Different File Size

File size (KB)	Key size (bits)	Computation Time of Hybrid (AES + ECC + SHA-256) [26]		Computation Time of Z-Buffer Algorithm	
		Encryption time(sec)	Decryption time(sec)	Encryption time(sec)	Decryption time(sec)
10	128	-	-	0.024	0.036
20	128	0.049	0.080	0.030	0.044
25	128	0.050	0.082	0.035	0.052
50	128	0.057	0.083	0.051	0.065
75	128	0.065	0.088	0.059	0.064
100	128	0.071	0.093	0.068	0.069
150	128	0.088	0.113	0.070	0.073
500	128	-	-	0.073	0.075
1000	256	-	-	0.078	0.081
1500	256	-	-	0.082	0.085
2000	256	-	-	0.087	0.091
2500	256	-	-	0.094	0.097
3000	512	-	-	0.099	0.102
3500	512	-	-	0.110	0.119

The approach suggested here gives a better result as compared to the existing ECC+AES+SHA-256 hybrid encryption with regard to encryption and decryption of text files. It is evident from Table 2, the execution time is decreasing with all sizes of the text files, which reflects on the computational efficiency and low-latency nature of the method with structured text data, specifically when faced with greater file sizes. Although hybrid approach continue to be used in widespread applications of general-purpose cryptography, but performance tends to become less effective when the size of files stored and used increases, or when there are high frequency encrypt-decrypt operations required. The text file is taken range of 10 KB to 3500KB in Table 2. For every file, the encryption and decryption times are evaluated. Key observations are as follows:

a) Encryption Time: A file size of 20 KB took 0.030 sec during encryption, while methods in [26] took 0.049 sec. Moving ahead, the proposed method consistently gives better results as the file size increases.

b) Decryption Time: Similarly with encryption method, decryption method works faster. For a file of 100 KB, the time is 0.068 se whereas methods in [26] took 0.071 sec.

c) Scalability: The proposed method shows a constant increase in computational time with file size increase but reduces the time as compared to the said reference, which shows the scalability and efficiency of the method.

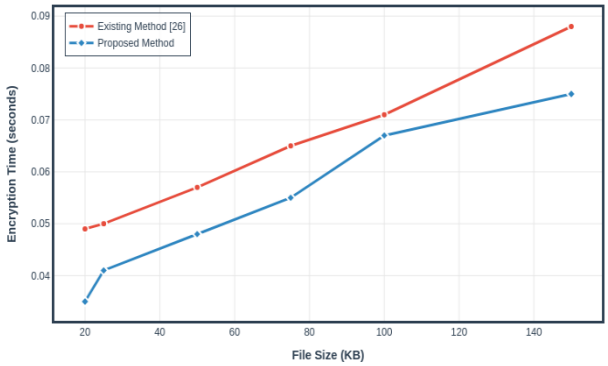

Fig. 4: Encryption Time Taken by Text Files.

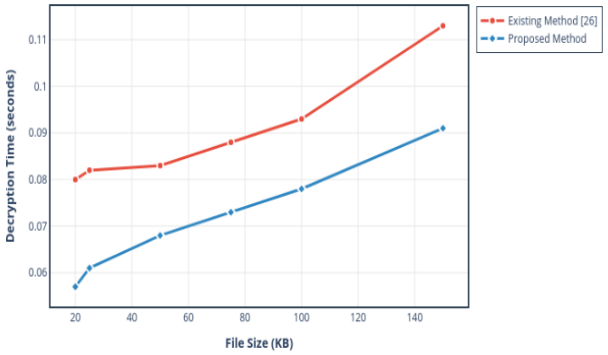

Fig. 5: Decryption Time Taken by Text Files.

Fig. 4 and Fig. 5 represent that the proposed method lines for encryption and decryption with methods available in [26] across all file sizes, showing its performance. The slope represents the steady and managed increase in the time with file size. Results show that the proposed method gives significant improvements in computation time. Hence, the requirement of method increased for applications that required frequent encryption and decryption of different file sizes.
The speed of Z-buffer-based cryptographic approach is higher than hybrid method and experiments demonstrate the necessary speed on the text data of a few kilobytes to several megabytes. The presented technique has proved to be considerably faster to utilize when encryption and decryption are concerned, yet maintaining high data integrity and minimal memory consumption, relative to files larger than 1 MB.
Z-buffer encryption can be optimized for ultra-low-latency applications by implementing lightweight versions of the algorithm with reduced computational complexity, using hardware acceleration like GPUs or FPGAs for faster parallel processing, and minimizing key computation time through pre-computed or reusable secure key structures. Additionally, adaptive compression before encryption and selective layer-based encryption can reduce data size and processing time, making it more suitable for real-time systems such as IoT or multimedia streaming.

4. Conclusion

The paper presents the classical use of the Z-buffer algorithm of computer graphics in a cryptographic system, which would allow both encryption and decryption of the data. The method has been implemented on large information at the early stage to confirm its effectiveness. The outcomes reveal a significant ability to process the complexity of data with use of index frequencies, ASCII-based operations to increase security. The approach also appears promising in terms of scalability, CPU usage and latency management. The layering facility of Z-buffer, places a new dimension on the cryptographic protection, which reduces the possible threats. The framework especially fits in the security of cloud storage, secure encryption, and decryption using blockchain, the security of the IoT devices. Despite the fact that the present work is focused on large data upto 1 GB, further studies are planned for large files more than 1 GB to test the reliability and flexibility with the help of images, sound and video files. Other improvements can be done though integration of compression mechanisms as well as the implementation of the procedure on distributed cloud systems. Future directions would also involve incorporating AI-augmented cryptography of intelligent key generation, GPU acceleration to reduce implementation latency of encryption and decryption, and modification to a post quantum context to overcome quantum security threats.

Acknowledgments

First author is grateful to University Grants Commission (UGC), New Delhi for providing the essential resources and financial support to the Babasaheb Bhimrao Ambedkar University, India for performing the said experiments and research work.

Funding

Presented work have no specific grant from any funding agency in commercial, public and not-for-profit sectors however first author is receiving the fellowship from UGC, New Delhi, India for performing said research work.

Conflict of interest

Authors declare no conflict of interest.

Author contributions

The first author performed the experimental work while the second author gives the idea of the research work and both the authors jointly wrote the said manuscript.

Ethics approval

Not Applicable

Data availability

The entire data and experimental work have been performed on own data sets.

References

[1] Liu P.P. and Wu Y.C (2018), "Research on Real-Time Graphics Drawings Technology in Virtual Scene". *In 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI),* pp. 1-6. IEEE. https://doi.org/10.1109/CISP-BMEI.2018.8633195.

[2] Bakshi N, Shivani S, Tiwari S and Khurana M.(2020), "Optimized Z-Buffer Using Divide and Conquer", *In Innovations in Computational Intelligence and Computer Vision,* pp. 41-47. https://doi.org/10.1007/978-981-15-6067-5_6.

[3] Mantler S. and Hadwiger M.(2007), "*Saving the Z-cull Optimization*", In ACM SIGGRAPH 2007 posters 2007; 7. https://doi.org/10.1145/1280720.1280729.

[4] Sakamoto Y, Takase M, Aoki Y.(2023), "Hidden surface removal using z-buffer for computer-generated hologram", *Practical Holography XVII and Holographic Materials IX* , Vol. 276-283. https://doi.org/10.1117/12.473824.

[5] Güney E., Bayilmiş C. and Cakan B. (2022), Corrections to "an implementation of real-time traffic signs and road objects detection based on mobile GPU platforms". *IEEE Access*.10, pp: 86191-86203. https://doi.org/10.1109/ACCESS.2022.3198954.

[6] Vasuki P., Shakin Banu A., Mohamed Mansoor Roomi S. and Maragatham G (2020), "Efficient target detection and classification of SAR images using Z-buffer convolutional neural networks". *InSmart Trends in Computing and Communications: Proceedings of SmartCom*; pp: 449-458. https://doi.org/10.1007/978-981-15-5224-3_45.

[7] Chen Z, Tagliasacchi A and Zhang H (2021), "Learning mesh representations via binary space partitioning tree networks", *Transactions on Pattern Analysis and Machine Intelligence.* 202129, Vol. 45, No.4, pp:4870-4881.

[8] Schiff, J. L. (2022). Topics in complex analysis. *Walter de Gruyter GmbH & Co KG*. 88.

[9] Sánchez-Ruiz, L. M., Legua, M., Moll-López, S., Moraño-Fernández, J. A., & Roselló, M. D. (2025), "The Exponential Versus the Complex Power ez Function Revisited", *Mathematics*, 13(8), 1306. https://doi.org/10.3390/math13081306.

[10] Alsammarraie, O., Al-Salami, Q., and Al-Salami, N (2024), " Transforming Coordinate Function Points into 2D or 3D Graphics: An Algorithmic Approach", *5TH International Conference On Communication Engineering and Computer Science*. https://doi.org/10.24086/cocos2024/paper.1562.

[11] Nielsen MA, Chuang IL. Quantum computation and quantum information. Cambridge university press; 2010.

[12] Lee, C., Hasegawa, H., and Gao, S. (2022), "Complex-Valued Neural Networks: A Comprehensive Survey", *IEEE/CAA Journal of Automatica Sinica,*Vol. 9, pp. 1406-1426. https://doi.org/10.1109/JAS.2022.105743.

[13] Victor, M., Praveenraj, D., R, S., Alkhayyat, A., and Shakhzoda, A (2023), "Cryptography: Advances in Secure Communication and Data Protection", *E3S Web of Conferences.* https://doi.org/10.1051/e3sconf/202339907010.

[14] Lv, Y (2021), "Data privacy protection based on homomorphic encryption", *Journal of Physics: Conference Series*; pp. 2037. https://doi.org/10.1088/1742-6596/2037/1/012129.

[15] Cheon J.H., Kim A., Kim M., Song Y.(2017),"Homomorphic encryption for arithmetic of approximate numbers", *In Advances in cryptology–ASIACRYPT 23rd International conference on the theory and applications of cryptology and information security*, pp.409-437. https://doi.org/10.1007/978-3-319-70694-8_15.

[16] Liu, S. (2022). Privacy Protection Revolution: Zero-knowledge Proof. *2022 International Conference on Data Analytics, Computing and Artificial Intelligence*, pp. 394-397. https://doi.org/10.1109/ICDACAI57211.2022.00084.

[17] Solomka, I., Liubinskyi, B., and Torshyn, V (2024), "Application of machine learning algorithms to enhance blockchain network security", *Mathematical Modeling and Computing.* https://doi.org/10.23939/mmc2024.03.893.

[18] Bennett CH, Brassard G. (2014), "Quantum cryptography: Public key distribution and coin tossing", *Theoretical computer science*. Vol. 560, pp.7-11. https://doi.org/10.1016/j.tcs.2014.05.025.

[19] Kumar J, Saxena V. Hybridization of Cryptography for Security of Cloud Data. International Journal of Future Generation Communication and Networking. 2020;13(4):4007-14.

[20] Saxena V and Kumar P. (2023), "Secure Transaction of Digital Currency through Fuzzy Based Cryptography", *Indian Journal of Science and Technology*. Vol.16, No.37, pp. 3148-58. https://doi.org/10.17485/IJST/v16i37.1453.

[21] Kumar P. and Saxena V. (2024), "Nested Levels of Hybrid Cryptographical Technique for Secure Information Exchange" , *Journal of Computer and Communications*. Vol. 12, No. 2, pp.201-10. https://doi.org/10.4236/jcc.2024.122012.

[22] Kumar P. and Saxena V. (2024), "Secure Transmission of Information through Hybrid Cryptographical Techniques", *TWIST*. Vol.19,No.2,pp.508-12.

[23] Gautam D. and Saxena V. (2023), "Secure exchange of IMSI number between sender and receiver", *Kepes*. Vol.21, No.3, pp:750-62.

[24] Gautam D, Rimer S. and Saxena V. (2023), "Secure access of folders and files after removal of duplicacy over the cloud", *International Journal of Computer Network and Information Security*. Vol.16, No. 1, https://doi.org/10.5815/ijcnis.2024.01.04.

[25] Gautam D and Saxena V. (2023), "Optimization of Storage of Cloud Servers Through Binary Search Algorithm", *7th Conference on Information and Communication Technology*, pp:1-6. https://doi.org/10.1109/CICT59886.2023.10455361.

[26] William P, Choubey A, Chhabra GS, Bhattacharya R, Vengatesan K. and Choubey S (2022), "Assessment of hybrid cryptographic algorithm for secure sharing of textual and pictorial content", *International conference on electronics and renewable systems*; pp:918-922. https://doi.org/10.1109/ICEARS53579.2022.9751932.

[27] Kinganga, J., Kasoro, N., Ramadhani, I., Musesa, A., Mabela, R., & Kenke, E. W. (2024). Literature review on data encryption based on elliptic curves and chaotic functions, and future prospects. *Journal Africain Des Sciences.*, *1*(2), 83–95. https://doi.org/10.70237/jafrisci.2024.v1.i2.10.

[28] Chen, A. C. (2023). Using Elliptic Curve Cryptography for Homomorphic Hashing. *In 2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES)*. pp. 1-5. IEEE. https://doi.org/10.1109/ICSSES58299.2023.10200709.

[29] Mahajan, H. B., and Junnarkar, A. A. (2023),"Smart healthcare system using integrated and lightweight ECC with private blockchain for multimedia medical data processing", *Multimedia Tools and Applications,* 82(28), 44335-44358. https://doi.org/10.1007/s11042-023-15204-4.

[30] Hosseingholizadeh, A., Rahmati, F., Ali, M., Damadi, H., and Liu, X. (2024), "Privacy-Preserving Joint Data and Function Homomorphic Encryption for Cloud Software Services", IEEE Internet of Things Journal, 11, 728-741. https://doi.org/10.1109/JIOT.2023.3286508.

[31] Morales, A. N., Bishwas, A. K., & Varghese, J. J. (2025). Quantum-enabled framework for the Advanced Encryption Standard in the post-quantum era. arXiv.

[32] Zala, D. K., and Khan, M. A. (2024, December). Novel Dual-Encryption and Compression Strategy: AES and Fernet for Secure Multimedia Transmission. In 2024 International Conference on IoT Based Control Networks and Intelligent Systems (ICICNIS) (pp. 239-244). IEEE. https://doi.org/10.1109/ICICNIS64247.2024.10823305.

[33] Yang, W., Xiang, X., Huang, C., Fu, A., & Yang, Y. (2023). MCA-based multi-channel fusion attacks against cryptographic implementations. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 13(2), 476-488. https://doi.org/10.1109/JETCAS.2023.3252085.

[34] Ramakrishna, D., & Shaik, M. A. (2024). A Comprehensive analysis of Cryptographic Algorithms: Evaluating Security, Efficiency, and Future Challenges. IEEE Access, 1. https://doi.org/10.1109/ACCESS.2024.3518533.

[35] Chahar, S. (2024). Exploring the future trends of cryptography. 234–257. https://doi.org/10.1201/9781003508632-16.