Journal of Engineering and Technology

ISSN 2180-3811 eISSN 2289-814X https://jet.utem.edu.my/jet/index

DOI: https://doi.org/10.54554/jet.2026.17.1.013

# A MULTI-DYNAMIC IMAGE STEGANOGRAPHY SYSTEM USING LEAST SIGNIFICANT BITS

O. A. Ayilara-Adewale*[1], O. E. Ojo[1], A. T. Adekunle[2] and P. O. Ajayi[3]

[1] Department of Information Technology, Osun State University, Oke Bale Street, Area, Osogbo 210001, Osun, Nigeria.

[2] Department of Computer Science, Osun State University, Oke Bale Street, Area, Osogbo 210001, Osun, Nigeria.

[3] Department of Computer Science, Oduduwa University Ipetumodu, P.M.B. 5533, Ile Ife, Osun, Nigeria.

*corresponding: oluwatobi.ayilara-adewale@uniosun.edu.ng*

**Article history:**
Received Date: 17 June 2025
Revised Date: 31 December 2025
Accepted Date: 5 May 2026

Keywords:
Decode, Encode, Least Significant Bit,

**Abstract**— The study aims to design and implement a multi-dynamic image steganography system capable of embedding multiple data formats (colored and grayscale images, and text) into a single cover image, enhancing flexibility and security in data transmission. The system was developed using Python programming language and the Least Significant Bit (LSB) technique. Ten (10) colored and grayscale images, alongside text data, were used for system testing. The system's imperceptibility was evaluated using the

| | |
|---|---|
| Steganography | Peak Signal-to-Noise Ratio (PSNR) metric. A dual embedding and decoding process was established to handle both text and image files, enabling simultaneous concealment. The average PSNR value obtained was 39.21 dB, indicating good image quality and minimal distortion. Results also showed that the system achieved higher PSNR when embedding text compared to images due to smaller file sizes. The results demonstrate that the proposed multi-dynamic approach improves versatility and maintains perceptual image quality compared to conventional monodynamic methods. The developed system ensures secure, flexible, and imperceptible data concealment, supporting the advancement of intelligent steganography systems. |

## I. Introduction

Internet usage has become an integral part of our daily lives, particularly in the realm of communication and information exchange through various media. Some of this information can be sensitive and needs to be securely transmitted without breach or alteration in the content of the shared information [1, 2]. To ensure a secure transmission of this information, some techniques can be adopted, such as encryption, IPsec, secure shell, two-factor authentication, data masking, and steganography.

Steganography refers to the art of covered or hidden writing". Steganography is a simple security method that originated from the Greek words "Stegos" which means "Cover" and "graphy" meaning "writing' hence defining it as "Covered writing [3, 4]. Steganography is a technique of hiding

information in another file, message, image or video through the use of any medium, such as a computer file, image, video or message [5]. The concept is to hide and transmit information so that it will not be easily identified because the coded information does not appear to be abnormal, i.e. its presence is undetectable by sight. Data or information hiding is pertinent to prevent data breaches or intruders because if this information can be obtained in a comprehensible or readable format, then it can easily be altered, disclosed or exploited to misrepresent an organization or individuals or facilitate an attack.

One of the identified techniques to securely transmit information by concealing it without being noticed or discovered is steganography [6, 7]. Steganography becomes more important as more people join the cyberspace revolution [8].Steganography encompasses a range of secret communication methods that conceal the message from being detected or revealed. There are two main steps in steganography: the first is embedding a secret message inside a cover image using a stego key, and the second step is extracting the secret message from the cover image using the stego key [9, 10]. This study developed a multi-dynamic image steganography using least significant bit.

## II. Review of Related Works

[11] developed a block-wise edge adaptive steganography scheme (BEASS) that targets textured regions, embedding three bits in edge pixels while maintaining statistical integrity, resulting in minimal distortion and resistance to multiple types of attacks. Additionally, [12] addressed the limitations of traditional methods, such as least significant bit (LSB) and pixel value differencing (PVD), by introducing pixel intensity ranges to enhance embedding capacity while preserving visual quality. The technique showed resilience against machine learning-based detection attacks. Furthermore, [13] applied a squeezeNet deep learning network to optimize cover image selection, enhancing visual

imperceptibility and security. Other studies, such as [9], combined techniques like Haar discrete wavelet transform with genetic algorithms to increase the robustness of stego images against attacks and enhance hidden data capacity.

Additional methods explored include hybrid systems that integrate various techniques for improved performance. For instance, [14] developed a matrix-based secret-sharing method to enhance security by distributing encrypted images across multiple data hiders. In contrast, [15] introduced an adaptive pixel value and pixel value difference method to improve embedding capacity in grayscale images using modified LSB substitution. The emergence of novel techniques, such as gradient-selective Bezier curves for text embedding in color images [12] and a blockchain-based framework for enhancing channel capacity [16], demonstrates a growing trend toward leveraging advanced technologies for secure steganography. However, despite these advancements, several constraints remain within existing systems, which this study aims to address.

The existing traditional steganography system allows the embedding of only one type of file, which can be either audio, image, or text, within a single image, which makes this a mono-dynamic system. This limits the ability to simultaneously manage different data formats, which reduces efficiency and flexibility in secure data transmission. Hence, this study proposes a multi-dynamic steganography system that can embed multiple types of data formats such as coloured and grayscale images, and text, within a single image, thereby improving versatility and security in the data transmission process. This study contributes to the advancement of the field of steganography, especially addressing the limitation of the conventional system that can only manage a single type of data format.

## III. Methodology

This section discussed the proposed multi-dynamic image

steganography system, which has two steps, encoding and decoding. Ten (10) JPEG-colored images and grayscale images were collected from online digital archives, which were used as cover and stego images. A user can enter any text of choice and an image for the encoding process. Thereafter, the user uploads the cover image, and the pixel maps of the images are loaded to determine their dimensions. This is important because the sizes of both images are compared, as the message image must be smaller than the cover image before further processes can be done. The system reads the Red, Green, and Blue (RGB) values of both images. It converts each to binary, after which it merges both binary values of the message image and cover image by substituting the four (4) least significant bits of the cover image for the four (4) most significant bits of the message image to create a new eight (8) bit binary image which will be saved as a newly created output image that retains the dimensions of the cover image.

For the text, the system splits the RGB template of the cover image into a red, green, and blue template, creating a 2D drawing template image that holds the text hidden. The system reads the RGB values of the template image and the cover image's red template. It converts them to binary before merging both to create a new template image, which will be substituted for the red template of the cover image, and merged with the green and blue templates of the cover image to create a new RGB image. After which, it will create a new image and write the new RGB template into this image; the output for both the image and text will be a single stego-image, this step-by-step is summarized in Algorithm 1 as shown in Table 1.

The image decoding process involves a user uploading the stego-image into the system, which loads the pixel map to get its dimension. After the dimension is gotten, the system reads the RGB value of the image, converts it to binary, extracts the last four (4) bits of the image, and adds four (4)

zeros (0000) to it to create a new eight (8) bit binary image which creates a new image to hold the output. The decimal result is written into the pixel of the new output image and displays the hidden image.

To decode the text message, the user uploads the stego-image, which the system splits its RGB template into red, green, and blue templates. The system then reads the RGB value of the red template and converts it to binary. The system extracts the last bits of the red template and writes them into a newly created output image, which holds the hidden text as a portable network graphics (PNG) image. The hidden text, the output, is displayed as a PNG image in the designated file, this is summarized in Algorithm 2 as shown in Table 2.

Table 1: Algorithm 1

| Image and Text Steganography Embedding Process | |
|---|---|
| 1 | Start |
| 2 | Input cover image img1 and secret message, consisting of a hidden image img2 and text txt. |
| 3 | Compare image dimensions: -<br>IF img2.size > img1.size:<br>*Error:* "Message image must be smaller than cover image."<br>ELSE:<br>Load pixel maps of both img1 and img2. |
| 4 | Load RGB template of img1 and split into channels: r_temp, g_temp, b_temp. |
| 5 | Get size and load pixel map of img1. |
| 6 | Create a 2D drawing interface imgdraw. |
| 7 | Draw text txt on imgdraw. |
| 8 | Create a new image output, retaining color mode and size of img1. |
| 9 | Read RGB values of img1 and convert them to binary form rgb1. |
| 10 | Read RGB values of img2 and convert them to binary form rgb2. |
| 11 | Merge the Most Significant Bits (MSB) of rgb1 and rgb2 to form a new 8-bit RGB value rgb. |
| 12 | Read RGB values of imgdraw and convert them to binary form rgb. |
| 13 | Convert the resulting rgb value to decimal form and store in the output image created in Step 8. |
| 14 | Merge r_temp, g_temp, and b_temp channels to create a new RGB image and store in the output image. |
| 15 | Save the output image as a stego-image (the final embedded image). |
| 16 | End |

Table 2: Algorithm 2

| | Image and Text Steganography Extraction Process |
|---|---|
| 1 | Start |
| 2 | Input stego-image img. |
| 3 | Load the pixel map and obtain the size of img. |
| 4 | Split the RGB color channels of img and extract the red channel as r_temp. |
| 5 | Get the size and load the pixel map of img. |
| 6 | Create a new image output, retaining the color mode and size of img. |
| 7 | Read the RGB values of img and convert them to binary form rgb. |
| 8 | Extract the last 4 bits of each rgb value and append 4 zeros to form a new 8-bit RGB value. |
| 9 | Convert the resulting rgb binary values to decimal form and store them in the corresponding pixels of the output image. |
| 10 | Convert r_temp values to binary. Extract the least significant bit (LSB) of every pixel and fill the pixels of the output image in black and white format:<br>If last bit == 0 → pixel(output) = white<br>Else last bit == 1 → pixel(output) = black |
| 11 | Output the hidden image and text in PNG format. |
| 12 | End |

Figure 1 depicts the multi-dynamic image steganography architecture consisting of the secret message and cover image phase, Stego image phase and Encryption and Decryption phase. The secret message (image or text or both) and cover image phase is the interface that allows a user to select the content to be hidden and the desired cover image to be used. Thereafter, the steganographic encoder embeds the secret message into the cover image using the standard LSB. The output from this steganographic encoder is the stego image, which serves as input for the steganographic decoder. The decoder phase is where the user extracts the hidden message from the cover image.

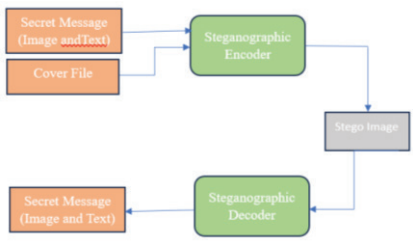



Figure 1: Multi-Dynamic Image Steganography Architecture

The operational definition of the components of multi dynamic image steganography architecture is discussed as follows:

- Secret Message:

This is the hidden message to be embedded into the cover image. The secret message can be in two formats: an image, text, or both. The essence of embedding is to prevent information from getting into the wrong hands, which can be used for malicious purposes. This system enables the proper encryption of this hidden message while retaining its quality.

- Cover Image:

The cover image can also be referred to as a decoy image. This image is used as a disguise for the existence of the secret message to prevent suspicion.

- Steganographic Encoder:

This is where the encryption process takes place. In this phase the encoder takes two inputs the payload (secret message) and the cover image and thereafter uses the LSB to modify the image's pixel values and gives a new image known as the stego image.

- Stego Image:

This is the modified image that contains the secret message within it using the steganographic encoder. It is the output of the embedded payload (secret messages, which can be image or text) into a cover image.

- Steganographic Decoder:

This is where the extraction of the secret message (payload) from the stego image that has been encoded using the steganography technique is done. The decoder does the reverse process of the steganographic encoder. The system identifies and extracts the hidden message without significantly changing the carrier medium.

## IV. Results Implementation

The algorithms presented in Section III were implemented using Python programming, making it easier for users to use steganography to embed secret images into a cover image. The cover image is then encoded and gives a stego-image, which can be decoded to retrieve the hidden file. The output of the system developed is demonstrated accordingly. Once the user initiates the software, the first view is the user's page, which presents the user with the encode and decode options, as depicted in Figure 2.


Figure 2: User Interface

The encoding allows the user to create a stego-image, while the decoding process retrieves the hidden file. The encode interface enables users to upload an image and text that need to be encrypted into a single cover image. After uploading, the user clicks "Embed" and "Save," as shown in Figure 3.

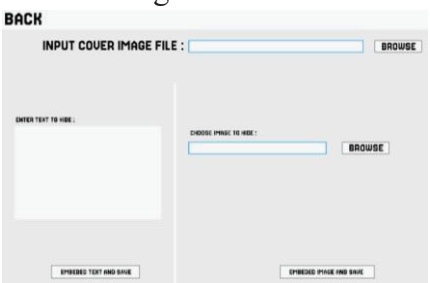



Figure 3: File Embedding Interface

Figure 4 depicts an interface where a user uploads the cover image, the text of choice and the image to be embedded. Thereafter, the user clicks on the "Save and Embed" button, which displays the cover image and the secret message. A pop-up indicating successful encryption is displayed in Figure 5. The output generates a single stego-image, which is saved in a designated file.

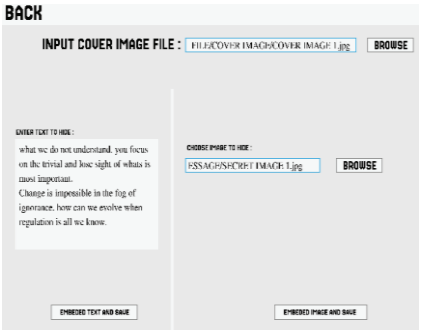



Figure 4: User input cover image and secret message (image and text)

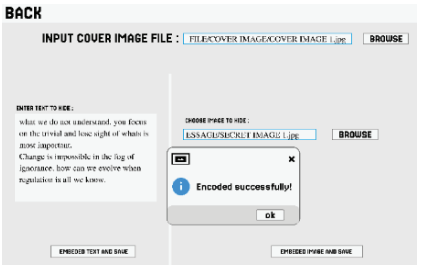



Figure 5: Successful Encoding Pop-up

The decoding interface is where the decryption of the secret message is performed; this is illustrated in Figure 6. The stego-image is uploaded, and the user clicks on the extract button to decode the text embedded in the image. A successful pop-up message shows the hidden text and image, as depicted in Figures 7 and 8. The extracted text comes in a PNG format.

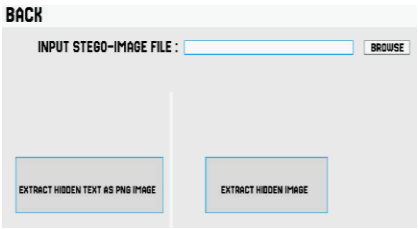



Figure 6: Decoding Interface


Figure 7: Successful Extraction Hidden Image Pop-up

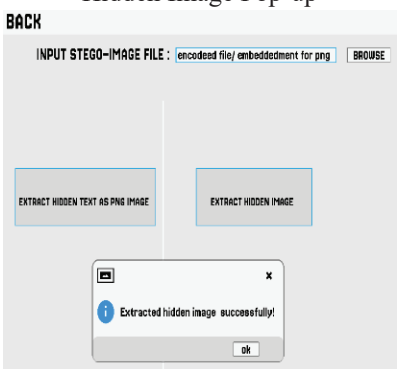



Figure 8: Successful Extraction of the Hidden Text Pop-up

## V. System Evaluation and Discussion

The system was tested with ten (10) images, both coloured and grayscale, and a desired text from the user. Similarly, a meticulous evaluation was performed to ascertain if the stego-images would experience any distortion throughout the embedding process of the cover image, but minimal distortion was observed. This result differs from that of [17, 18] and [11] who observed distortion in their evaluations when embedding images with a cover image.

The visual evaluation of the cover image and corresponding stego image revealed minimal distortion. The developed system was evaluated using the peak signal-to-noise ratio (PSNR), which is the ratio between the possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Table 3 shows the PSNR for 10 image sets in colored, grayscale, and text message.

Table 3: Image Steganography Performance

| Image | PSNR Value (dB) | | |
|---|---|---|---|
| | Colored (cover image and stego-image) | Grayscale (cover and stego-image) | Text (secret message and stego-image) |
| Set 1 | 30.17 | 30.68 | 56.48 |
| Set 2 | 31.65 | 30.50 | 54.50 |

| | | | |
|---|---|---|---|
| Set 3 | 32.64 | 31.45 | 56.79 |
| Set 4 | 31.54 | 30.57 | 53.87 |
| Set 5 | 31.84 | 29.95 | 52.95 |
| Set 6 | 32.45 | 31.37 | 55.06 |
| Set 7 | 33.20 | 30.53 | 55.35 |
| Set 8 | 29.99 | 29.91 | 54.87 |
| Set 9 | 30.90 | 32.43 | 53.65 |
| Set 10 | 31.34 | 31.86 | 57.97 |
| Total | 315.72 | 309.25 | 551.49 |

The mean of PSNR value for full colored image, grayscale image and image sets used in hiding text as shown in Equation (1) to (3).

$$Mean = \frac{TOTAL\ PSNR\ VALUE}{10} = \frac{315.72}{10} = 31.57\ dB \quad (1)$$

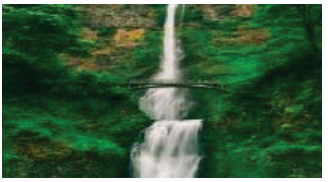
$$Mean = \frac{TOTAL\ PSNR\ VALUE}{10} = \frac{309.25}{10} = 30.92\ dB \quad (2)$$

$$Mean = \frac{TOTAL\ PSNR\ VALUE}{10} = \frac{551.49}{10} = 55.15\ dB \quad (3)$$

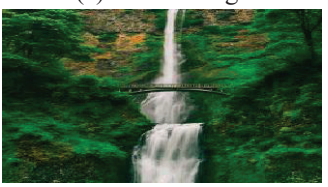
Therefore, the average PSNR value for the entire system is calculated Equation (4).

$$Average\ PSNR\ for\ the\ system = \frac{31.57+30.92+55.15}{3} = 39.21\ dB \quad (4)$$

The PSNR value of 39.21 dB obtained from the system evaluation further indicates that the system's average PSNR value falls within the usual range for 8-bit images, indicating good imperceptibility and supporting the significance of this study. This is supported by the reports of [19] who revealed that good image quality must have values between 30 dB and 40 dB. Equally, our result differs from that of [20], who observed PSNR values exceeding 40 when using an MLSB approach to evaluate the system they developed, which employed random pixel selections. However, the results show that the system achieved higher PSNR values when hiding text than when hiding images, due to the smaller size of text files compared to image files. Figure 9 shows the sample of visual comparison between cover and stego-image.



(a) Cover Image



(b) Stego Image

Figure 9: Visual Comparison between Cover and Stego Image

## VI. Conclusion and Future Works

This research develops a multi-dynamic image steganography system. The system is designed based on the foundation of steganography techniques, embedding images in RGB, grayscale, and text and retaining their image quality after decoding. The system was implemented using Python programming language and tested using 10 images of colored, grayscale, and random texts. The study's results established that the multi-dynamic system exhibits good imperceptibility and higher PSNR values when embedding text than images, primarily due to the smaller file size of text files compared to image files.

Future studies can focus on using private keys for encryption and decryption, respectively, to enhance the system performance. Similarly, compression of image files can be considered to achieve a higher PSNR value when embedding.

## VII. References

[1] I. Keshta and A. Odeh, "Security and privacy of electronic health records: Concerns and challenges," *Egyptian Informatics Journal*, vol. 22, no. 2, pp. 177-183, 2021.

[2] S. Shukla, J. P. George, K. Tiwari, and J. V. Kureethara, *Data ethics and challenges*. Springer, 2022.

[3] M. Idakwo, M. Muazu, E. Adedokun, and B. Sadiq, "An extensive survey of digital image steganography: State of the art," ATBU Journal of Science, Technology and Education, vol. 8, no. 2, pp. 40-54, 2020.

[4] A. Zakaria, "*Batch steganography and pooled steganalysis in JPEG images*," Université Montpellier, 2020.

[5] N. Arora, "Types and tools of steganography," *International Journal for Research in Applied Science and Engineering Technology*, vol. 10, no. 6, pp. 2049-2053, 2022.

[6] S. Ghoul, R. Sulaiman, and Z. Shukur, "A review on security techniques in image steganography," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 6, 2023.

[7] M. Shelke, N. Kalyankar, M.Adagale, P. Bhandare, and A. Batule, "Privacy Preservation Using Data Hiding and Data Concealing with Steganography and Cryptography," in *International Conference on Power Engineering and Intelligent Systems (PEIS), 2023*: Springer, pp. 169-178.

[8] A. K. Babando and B. M. Ahmad, "Data security using steganography," *LC International Journal of STEM*, vol. 3, no.4, pp. 12-24, 2022.

[9] A. ALabaichi, M. a. A. A. K. Al-Dabbas, and A. Salih, "Image steganography using least significant bit and secret map techniques," *International journal of electrical & computer engineering* vol. 10, no. 1, 2020.

[10] R. Bansal and N. Badal, "A novel approach for dual layer security of message using Steganography and Cryptography," *Multimedia Tools and Applications*, vol. 81, no. 15, pp. 20669-20684, 2022.

[11] D. Laishram and T. Tuithung, "A novel minimal distortion-based edge adaptive image steganography scheme using local complexity: (BEASS)," *Multimedia Tools and Applications*, vol. 80, no. 1, pp. 831-854, 2021.

[12] S. N. Mohammed, "Color Image Steganography Using Gradient Selective Bezier Curves," *Iraqi Journal of Science*, pp. 3625-3641, 2023.

[13] C. Nkuna, E. Esenogho, R. Heymann, and E. Matlotse, "Using Artificial Neural Network to Test Image Covert Communication Effect," *Journal of Advances in Information Technology*, vol. 14, no. 4, 2023.

[14] Z. Hua et al., "*Matrix-based secret sharing for reversible data hiding in encrypted images*," IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 5, pp. 3669-3686, 2022.

[15] A. G. Chefranov and G. Öz, "Adaptive to pixel value and pixel value difference irreversible spatial data hiding method using modified LSB for grayscale images," *Journal of Information Security and Applications*, vol. 70, p. 103314, 2022.

[16] Z. Chen, L. Zhu, P. Jiang, J. He, and Z. Zhang, "*A Generic Blockchain-based Steganography Framework with High Capacity via Reversible GAN*," in IEEE INFOCOM 2024-IEEE Conference on Computer Communications, 2024: IEEE, pp. 241-250.

[17] S. Gaur, S. Chaturvedi, S. Gupta, J. Mittal, R. Tanwar, and M. Goswami, "*Image Distortion Analysis in Stego Images Using*

*LSB*," in Proceedings of Third International Conference on Computing, Communications, and Cyber-Security: IC4S 2021, 2022: Springer, pp. 667-679.

[18] Y.-L. Pan and J.-L. Wu, "Rate-distortion-based stego: A large-capacity secure steganography scheme for hiding digital images," *Entropy*, vol. 24, no. 7, p. 982, 2022.

[19] F. M. A. Albkosh, A. S. Mohamed, A. A. Albakoush, and A. A. Albkush, "*An Enhanced Method to Secure the High Embedding Capacity Steganography Based on LSB Indicator*," in 2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA), 2024: IEEE, pp. 494-499.

[20] O. M. Al-Shatanawi and N. N. El Emam, "A new image steganography algorithm based on MLSB method with random pixels selection," *International Journal of Network Security & Its Applications*, vol. 7, no. 2, p. 37, 2015.