Enhancing Steganography by Image Segmentation and Multi-level Deep Hiding

Alaa Al-Ahmad, Omar Saad Almousa and Qusai Abuein

Computer Science Department, Jordan University of Science and Technology, Jordan

Abstract: In this paper, we present Modified Deep Hiding Extraction Algorithm (MDHEA) that is a steganography algorithm with Multi-Level Steganography (MLS) and color image segmentation. Through experimental results, MDHEA shows improvement in the results of previous works by securing encrypted secret data against attacks. We use segmentation to choose the appropriate segment, pass it on the cover image, calculate the value of the change at the pixel of the segment and select the best segment and its location in the cover image based on the least effect. MDHEA applies multi-level steganography to hide the confidential data in color images to ensure the integrity of the hidden data and obtain the largest volume of hidden data without causing any distortion in the stego image. To reduce distortion in the cover image due to hiding a large amount of secret data and obtaining a high-quality stego image after hiding the secret data, we implement the Blue Smoothing Algorithm (BSA) to achieve smoothing the largest possible number of pixels in the image.

Keywords: steganography, multi-level steganography, information hiding, encryption, smoothing, image segmentation.

1. Introduction

Transmitting information through local and international networks and computers has become a daily routine. It cannot be dispensed with because of its obvious effect in facilitating the demands of our modern life [1]. Such information should be transmitted effectively and securely.

Therefore, there is a need to rely on a method or technique to conceal and hide data within a digital medium. It is necessary to hide any clue about exchange of secret information. Such that only concerned persons are aware of this secret exchange of information. The fundamental idea of hiding information is to prevent unauthorized users from noticing and recognizing hidden data. Later, information hiding was adopted as a method to send information privately.

Researchers have worked thoroughly to create many methods for dealing with secret data and hiding it in the stego file successfully and came up with efficient algorithms related to hiding secret data with high complexity and difficulty to retrieve the hidden data. Multi Levels Steganography (MLS) technology makes it more difficult to access hidden data compared to Single Level Steganography (SLS) technology [2]. When thinking of a strong data security system, two major issues to be considered, hiding efficiency, and hiding payload. The first thing that the researchers tried to develop is to match the stego file with the cover file, so it is not discovered by attackers to view the secret data. The second thing is to develop a method that achieve as high data hiding efficiency as possible in the cover file. The more efficient the hidden data is; the less data load will be [3].

Many algorithms have been used for steganography, some of which are complex because they need a long time to hide secret data, while others are characterized by simplicity and flexibility and least complexity of hiding secret data as in the Least Significant Bit (LSB) [4].

In [5], authors classified multimedia into five types: text, image, audio, video, and protocol. The first four types involve hiding information in the file format, while the fifth one is more extensive and includes the use of a platform or a contact protocol to hide information.

Other studies suggest a technique to hide secret data, relying on two types of color and grayscale images for a deep hiding and extraction algorithm (DHEA). This technique was implemented using multi-level Steganography (MLS) [6]. They worked on the modified least significant bit (MDLSB) to include secret data in the cover image used at each level. To avoid distortion in the stego image at the last level, the smoothing algorithm was used to reduce the noise generated by the stego image and to achieve homogeneity between the color components in one pixel. To increase security, it relied on the last level on segmentation at the segment level and at the pixel level to disperse data and hide it securely. The number of levels was randomly determined, with each level using a color image, and the image size is expanded when moving from one level to the next level. An appropriate pixel randomization approach is applied to hide secret data at each level [7].

This paper proposes an enhancement on data security by using an algorithm based on deep hiding using a color image. We use a segmentation method that picks the least noisy segment by determining the best segment in a cover image. Every segment is taken and passed on the cover image at all available locations. Each time the number of modified bits per pixels is calculated to select the lowest one. The best segment suits the size of the secret massage to be hidden without affecting the quality of the cover image. Then distribute the secret message over the pixels of the selected segment in a non-sequential manner. After that, exploit each byte by choosing the most appropriate bit to hide data, considering not increasing the noise in the stego image, and maintaining the best balance between the amount of payload and the image quality after hiding data.

2. Materials and Methods

MDHEA hides secret data inside color images through multi-levels. MDHEA aim is to enhance the security of secret data and payload capacity, shown in Figure 1. The fundamental idea is to pick the least noisy segment by determining the best segment in a cover image to hide the data in it. The proposed algorithm (MDHEA) was implemented on a wide range of secret data while preserving it from attackers using visual and statistical attacks. The proposed algorithm (MDHEA) performs deep hiding and multi-level steganography (MLS). MDHEA has two phases, the hiding phase (MDHA) and the extracting one (MDEA).

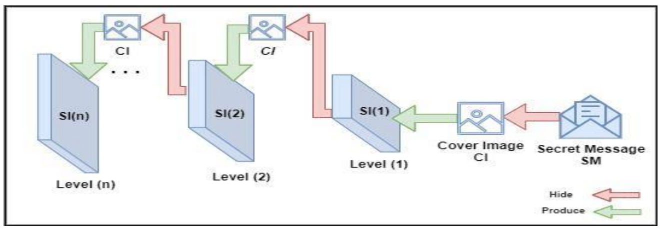


Figure 1. Modified deep hiding based on MLS

At each level of MDHEA, a set of procedures are implemented. First, MDHEA prepares the secret message. It selects the best segment in the cover image to hide the secret data considering the sizes of the secret data and the cover image to increase security and payload capacity. Then, MDHEA proceeds with compressing the secret message (Sm) and it at the first level to obtain (CSm). The compression is also applied for the rest of the levels of the stego image (Si) to obtain (CSi).

The compression is performed using the Deflate algorithm [8]. After the compression step, a symmetric encryption algorithm is used to obtain (ECSm) for the secret message. Then, encryption is applied for the rest of the stego image to obtain what we call (ECSi). The encryption is performed using AES [9]. In a nutshell, to prepare the secret message we apply the following steps (algorithm 1):

1. Read the secret message (Sm)
2. Compress Sm using the Deflate algorithm to produce CSm
3. Encrypt CSm using the AES algorithm to produce ECSm

The segmentation method is used to find the best segment for including the secret message. It starts by selecting pixel locations within the segment to hide the secret data using (4 bits) per pixel distributed across the three-color components (RGB) while maintaining the image quality. The deep hiding method requires in each level a cover image (CI) to hide the secret message (Sm) that has been compressed and encrypted to produce the stego image (SI), where the output of the level (n) is (SI^n) is an entry to the level (n +1). Figure 2 illustrates the phases to obtain deep hiding of data for n levels.

The proposed image segmentation algorithm is based on the adaptive non-uniform segmentation of a cover image (CI) and implemented at all levels.

We apply the following steps to display the suggested image segmentation to get available segments (algorithm 2).

1- Determine the size of the secret message.

2- Select the segment relative to the size of the cover image and the size of the secret message.

3- The segment begins to increase gradually until it completes the full cover image, where the value of each segment within the cover image is calculated and its movement is sequential.

4- The value of (Segx) remains fixed and the value of (Segy) increasing by one until reaching the last point of (Segy). After finishing all points in (Segy) increases the value of (Segx) one value and then repeats it until reaching the last point of (Segy) and so on.

After getting an available segment, every segment is taken and passed on the cover image (CI) at all available locations and each time the number bits per pixel (Nbpp) to be modified is calculated. The number of bits is calculated which will be adjusted for each segment in each location and then the lowest value will be taken, and the following data will be installed (value, Seg X starting location, and Seg Y starting location). The best starting point for the segment is specified in the cover image (CI) which represents (PerPx, PerPy), where (PerV) represents the lowest value for the segment. We call the previous steps (algorithm 3).

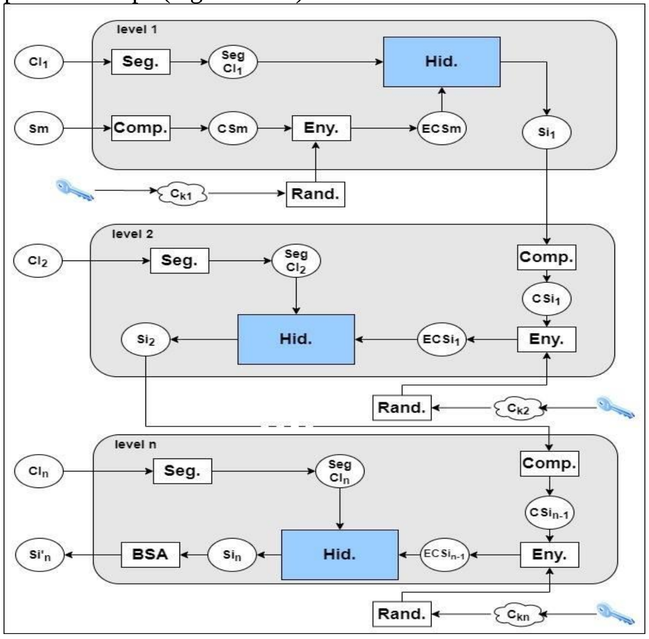


Figure 2. Phases to obtain deep hiding of data for n levels

The line marked in the table included in Figure 3 marks the best segment chosen in the cover image. After selecting the best segment and its location in the cover image according to the previous algorithm, hiding secret data begins. If the list of segments = 0, then the size of the cover image is small and is not enough to hide the secret message. After filling all values for all segments, we select the perfect segment as the smallest bit changed counter RGB, and we select iWx as the perfect start location with X coordinate (PerPx) and iHy as the perfect start location with Y coordinate (PerPy).

The proposed MDLSB algorithm uses uniform numbers of bits for hiding, it uses (4 bits) per pixel distributed over the three-color components (RGB), while maintaining the stego image quality and increasing payload. The pixel locations within the segment were chosen according to algorithm 3, ensuring that there are no sequences and no duplication of pixels' locations. After selecting the pixel locations, the hiding procedure starts (4 bit) in each pixel within the three-color components (R, G, B) according to the following steps:

1- Replace four bits to every pixel.

2- Get the stego image.

3- Generate the key with random symbols and encrypt them with symmetric.

According to the previous steps (algorithm 4), the stego image is obtained and a key is generated containing (SegX, PerPx, KeyX, SegY, PerPy, KeyY, KeyToEncript) between it symbols and numbers and their encryption of the type of symmetric encryption, which will be used to decrypt and extract the secret data. In the last level, after bit replacement algorithm finishes, it produces the final stego image. The last stego image is improved by using what we call the blue smoothing algorithm (BSA) to reduce distortion. The pixel preparation aims to get the closest color value to its original value before hiding the secret data and the change in the original pixel value. As it is known, pixels consist of three-color components: red, green, and blue.

In (MDHEA), four bits are distributed and changed on these components, one in the red, and another in the green, the amount of change is very simple and does not affect the value of the color component. As for the blue component, it is replaced by two bits, so the amount of change is clear. As a result, the blue smoothing algorithm (BSA) is applied to it at the closest value to the color of its original value.

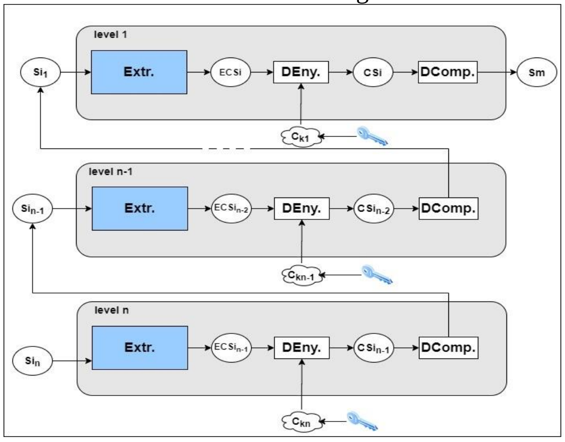


Figure 3. Modified deep extracting (MDEA) architecture

This is done by using the bits that were not used to hide and manipulate the data to bring the color value closer to the closest possible value to its original value, according to algorithm 5 as below:

1- Get the pixels that have their value been a change in the stego image compare with the cover image.

2- Get the perfect value for the pixel of stego image that closes to the value of the pixel in the cover image.

3- Replace the perfect value that gets from step two to the stego image and generate a smoothed stego image.

In Modified Deep Extracting Algorithms (MDEA) and after the recipient receives the stego-image, which is the same as the smooth stego image, because the change done by using the bits that were not used to hide and manipulate the data, it is provided with the agreed parameters for each level, these parameters are (number of levels (n) and cipher key (Ck)). This parameter has been used to extract secret message (Sm) and stego image (SI) from each level, then decrypt, decompress, and obtain the next secret message (Sm) and stego image (SI). These steps are applied until the required secret message has been combined, as depicted in Figure 4. According to algorithm 6:

1- From the last layer, we get the stego image.

2- From the file of keys decrypt all keys and produce a list of keys.

3- Using the list of keys, we decrypt the last stego image and decompress it, repeat this step until reach first layer.

4- Get the secret message from the first layer.

3. Experiments and Results

This section explains the experiments performed, its results and discussion. The data used to perform the experiments are set of color images taken from the dataset (UCID v2) [10] that is a well-known and widely used set of different color images used in the field of image processing. Comparisons were made with other works in the field of steganography through the size of hidden data and the noise that results from data hiding. The test was based on four-color images of size (512 x 512) shown in Table 1.

The performance of our proposed algorithm MDHEA in comparison to algorithms proposed in [6] and [7] is shown in Table 2. Table 2 shows experimental results calculated using PSNR for sample images with size (512×512) using (MDHEA). The results show a higher PSNR value relative to the previous techniques [6] and [7]. In addition to increasing the security and payload of data in the proposed approach for using segmentation in the cover image at each level. In addition, the MDLSB used takes advantage of as many bits as possible per pixel within the sector.

Table 2 shows the results of the proposed algorithm (MDHEA), as it is compared with the techniques proposed in [6] and [7] using deep hiding technique. The work has been conducted to find a PSNR value on a different payload capacity for the selected image sample.

Table 1. (.bmp) images of the same size used for testing MDHEA

Image				
Name	Baboon	Peppers	Lena	Barbara
Size	512×512	512×512	512×512	512×512

The results show that for an image at a payload capacity (2 x 104) bits, the efficiency of MDHEA is better than other algorithms in [6] and [7]) by approximately 10.62%, and 8.15% respectively. While for Lena image it is better by approximately 7.97%, and 7.12% respectively. MDHEA result is better by approximately 14.53%, and 12.25% respectively for Baboon image, and better by approximately 7.12%, and 3.99% respectively for Barbara image.

When calculating the PSNR at a maximum payload capacity (10.5 x 104) bits for the peppers image, MDHEA is better than the previous works mentioned above by approximately 12.66%, and 11.41% respectively. When the Lena image has a maximum payload capacity of (15 x 104), MDHEA is better than previous works [6] and [7] by approximately 10.92%, and 9.68% respectively.

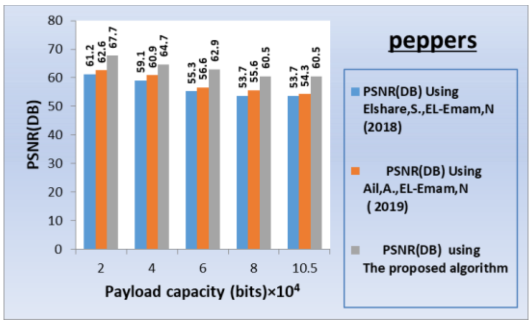
MDHEA result for the maximum payload capacity equals to (5.6×104) bits of the Baboon image achieves results better than the previous works mentioned above by approximately 18.16%, and 17.50% respectively. While when the maximum payload capacity equals to (12.5×104) bits for Barbara image, MDHEA results are better than the other algorithm by approximately 8.94%, and 8.35% respectively.

Figure 5 shows the comparison of the PSNR metrics value using different images, peppers (A), Lena (B), Baboon (C) and Barbara (D) respectively.

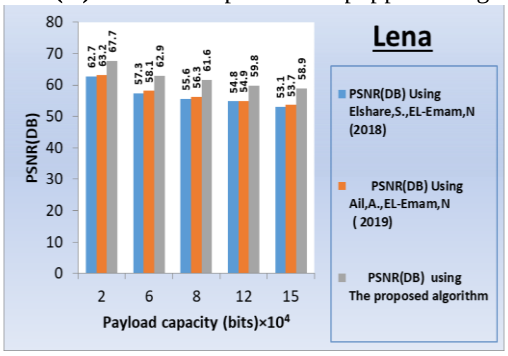
On the other hand, another list of images of the same type (.bmp) with different size have been used for further testing the MDHEA. The images are shown in Table 3.

Table 2. (.bmp) images of different sizes used for testing MDHEA

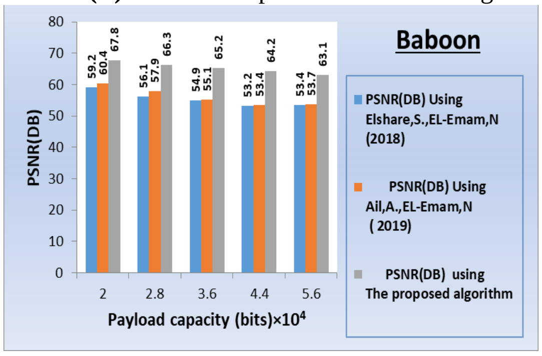
Image				
Name	Koala	Onion	People1	Tulips
Size	1024×768	198×135	431×359	276×183



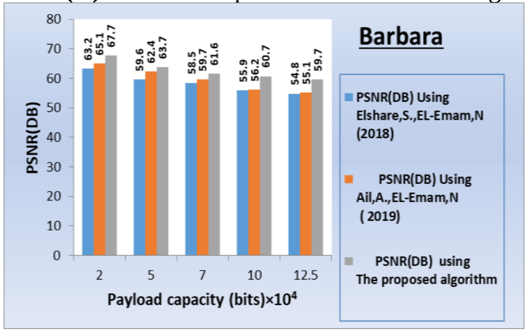
(A) PSNR comparison for peppers image



(B) PSNR comparison for Lena image



(C) PSNR comparison for Baboon image



(D) PSNR comparison for Barbara image
Figure 4. PSNR Comparison for different images

Table 4 shows the performance of the proposed algorithm using different images size and different testing metrics, namely MSE, SNR and PSNR.

Table 3. Performance of the proposed algorithm using different images size and different testing metric

Images name	Images size	MSE	SNR	PSNR
Koala	1024×768	0.0821	37.9177	58.9873
Onion	198×135	0.0012	47.3789	77.2156
People1	431×359	0.0002	51.7228	85.0204
Tulips	276×183	0.0006	50.3861	80.1583

The performance of the blue smoothing algorithm (BSA) used in the proposed algorithm (MDHEA) was evaluated by performing the test on the peppers, Lena, Baboon and Barbara image in Figure 5A; 5B; 5C; 5D, respectively. The result in Table 5 shows the efficiency of using the blue smoothing algorithm (BSA) within the proposed algorithm (MDHEA) in terms of distortion values using the PSNR and MSE metrics.
Figure 6 shows a comparison between MDHEA with BSA and without BSA for the four images: peppers (A), Lena (B), Baboon (C) and Barbara (D) in terms of MSE and PSNR values.
The steganalysis techniques are the inverse methods of steganography techniques, where these techniques work to analyze and process data from the digital media used to hide that data. The primary goal of the steganalysis tool is to determine if the messages have data inside them or not and recover hidden information if possible. A set of steganalysis tools are available to detect the presence of hidden information with the stego image like StegDetect, StegSecret, jeeps, shagbark and stages V2.1. [11]. These tools measure the power of the algorithm used to hide data.
A test on the used algorithm MDHEA has been conducted by applying more than one of steganalysis tools. StegDetect is a tool that makes detecting steganographic data in JPEG images. If no hidden data is found it will give the "negative" result. StegSecret is a software makes the detection of hidden data in various digital media. JPSeek is a software that helps the user to determine if the jpeg image has inside it a data or not [12]. StegSpy is a tool that detects steganography and the program used to hide the message, if no hidden data is found

it will give the “Sorry no Steg found” result [13]. We analyzed five different steganalysis tools in order to decide which one to use for evaluating MDHEA algorithm. We summarize the results in Table 6.

Table 4. The proposed algorithm (MDHEA) performance with (BSA) algorithm and without (BSA) algorithm

Images 512x512	Payload capacity (bits)×10^4	Before using Blue Smoothing algorithm (BSA)		After using the Blue Smoothi	
		MSE	**PSNR**	**MSE**	**PSNR**
	2	0.0108	67.7959	0.0061	70.2768
	4	0.022	64.7044	0.0125	67.1411
	6	0.033	62.9425	0.0187	65.3962
	8	0.0576	60.5239	0.0325	63.006
Peppers	10.5	0.0572	60.5497	0.0327	62.9803
	2	0.0109	67.752	0.0063	70.1247
	6	0.0331	62.9218	0.0191	65.2982
	8	0.0444	61.6539	0.0252	64.1113
	12	0.0669	59.8742	0.0383	62.2965
Lena	15	0.0827	58.9542	0.0474	61.3712
	2	0.0107	67.8101	0.0062	70.1686
	2.8	0.0151	66.3199	0.0091	68.5797
	3.6	0.0194	65.2362	0.0112	67.6268
	4.4	0.0246	64.2071	0.0139	66.6746
Baboon	5.6	0.0312	63.178	0.0174	65.7103
	2	0.011	67.7087	0.0063	70.1247
	5	0.0272	63.7694	0.0158	66.127
	7	0.0442	61.6731	0.0253	64.0838
	10	0.0551	60.7168	0.032	63.0703
Barbara	12.5	0.0696	59.7004	0.0401	62.0902

Table 6. Image Type of Steganalysis Tools

Steganalysis Tools	**Cover Image Type**		
	JPEG	BMP	**Others**
StegDetect	Yes	-	**-**
StegSecrect	Yes	-	**-**
JPSeek	Yes	-	**-**
StegBreak	Yes	-	**PNG**
StegSpy	**-**	**Yes**	**-**

Among the available tools for steganalysis, we have chosen The StegSpy steganalysis since MDHEA deals with BMP images, and StegSpy is the only tool that deals with the BMP images type. We fed the StgSpy tool with different MDHEA BMP images that are chosen to test the proposed approach from Table 1 and Table 3. The results of feeding StegSpy tool with those images are shown in Table 7.

The results show that none of the used tools uncovered the existence of steganography that we have in the images. This is basically a strong indication that our information hiding algorithm works just fine.

Table 7. The proposed algorithm (MDHEA) performance test

Images name	Images size	Steganalysis Tool
		StegSpy
Koala	1024×768	No Steg found
Onion	198×135	No Steg found
People1	431×359	No Steg found
Tulips	276×183	No Steg found
Baboon	512×512	No Steg found
Peppers	512×512	No Steg found

4. Conclusion

Steganography is one way to keep data secure from people who do not have access to it. This is done by preserving the stego image quality from any distortion, after hiding the secret data so it cannot be distinguished from the cover image.

In this work, a modified deep hiding extracting algorithm (MDHEA) to hide secret data at multi-levels with different sizes of color images used at each level in order to increase the level of security has been proposed. In addition, data compression was implemented to enable it to hide as much secret data as possible, then encrypt it to increase the level of security, it was relied on the cover image segmentation and choosing the best segment to hide the data according to its size in each level. At the last level, the proposed blue smoothing algorithm (BSA) was applied to reduce distortion in the final stego image.

The experimental results were compared with previous works. We showed that the stego image cannot be distinguished from the cover image. One of the major demands of the good steganography method is to provide good PSNR and MSE values. Our proposed method provides high PSNR and low MSE values when compared with other methods. We achieved an increase in the loaded capacity of the hidden data and the high level of security. We also used different steganalysis tools to check the strength of our proposed stganograpghy. The used tool was not able to uncover any existence of steganography that we have in the images using MDHEA. This is basically a strong indication that our information hiding algorithm works just fine.

References

[1] S. Khosla, P. Kaur, "Secure data hiding technique using video steganography and watermarking," International Journal of Computer Applications, Vol. 95, No. 20, pp. 7-12, 2014.

[2] N. S. Sikarwar, N. S. "An integrated synchronized protocol for secure information transmission derived from multilevel steganography and dynamic cryptography," International Journal of Computer Science and Telecommunication, Vol. 3, No. 4, pp. 31-36, 2012.

[3] R. J. Mstafa, K. M. Elleithy, "An efficient video steganography algorithm based on BCH codes," ASEE, 2015.

[4] N. Raftari, A. M. E. Moghadam, "Digital image steganography based on assignment algorithm and combination of DCT-IWT," Fourth International Conference on Computational Intelligence, Communication Systems and Networks, pp. 295-300, 2012.

[5] G. Latika, Y., "A Comparative Study and Literature Review of Image Steganography Techniques," International Journal of Science, Technology, and Engineering, Vol. 1, No. 10, pp. 238-241, 2015.

[6] S. Elshare, N. N. El-Emam, "Modified Multi-Level Steganography to Enhance Data Security," International Journal of Communication Networks and Information Security, Vol. 10, No. 3, pp. 509, 2018.

[7] A. Ali, "Improved Deep Hiding / Extraction Algorithm to Enhance Payload Capacity and Security Level of Hidden Information," MSc thesis, Philadelphia University Research. 2019.

[8] S. Oswal, A. Singh, K. Kumari, "Deflate compression algorithm," International Journal of Engineering Research and General Science, Vol. 4, No. 1, 2016 pp. 430-436.

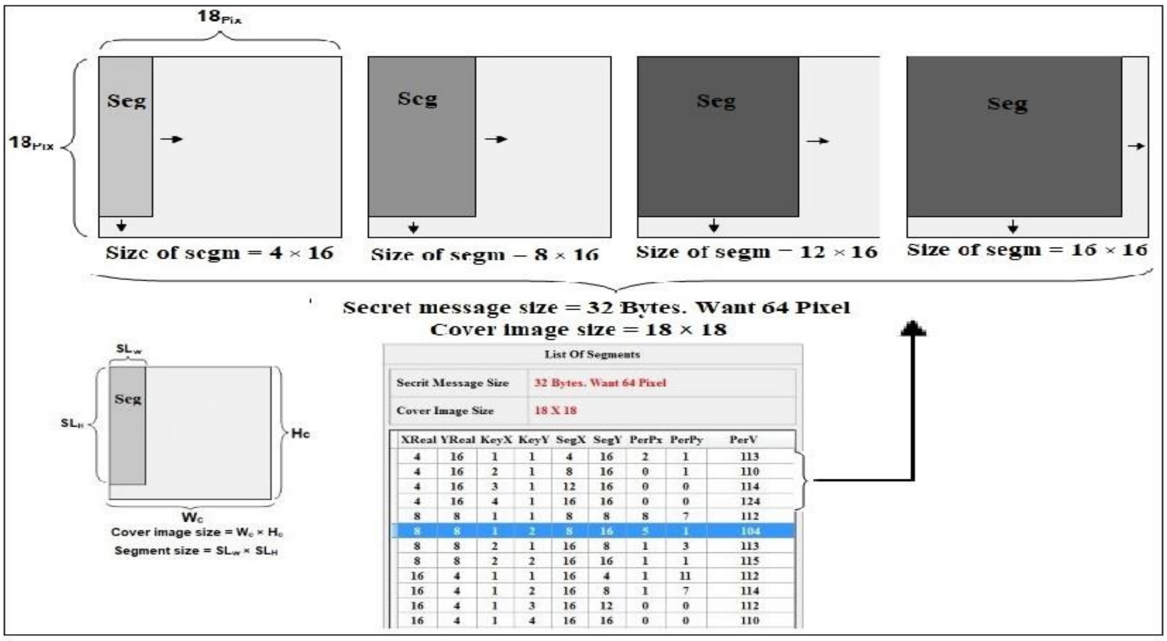
[9] A. K. Mandal, C. Parakash, A. Tiwari, "Performance evaluation of cryptographic algorithms: DES and AES," IEEE Students' Conference on Electrical, Electronics and Computer Science, pp. 1-5. IEEE, 2012.

[10] G. Schaefer, M. Stich, M., "UCID: An uncompressed color image database," Storage and Retrieval Methods and Applications for Multimedia. Vol. 5307, pp. 472-480, 2003.

[11] N. F. Johnson, S. Jajodia, "Exploring steganography: Seeing the unseen," Computer, Vol. 31, No. 2, pp. 26-34, 1998.

[12] L. Rathika, B. Loganathan, M. P. Scholar, T. Nadu, T. Nadu, "Approaches and methods for steganalysis–A survey," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 6, No. 6, pp. 433-438, 2017.

[13] J. L. Duffany, M. D. Velez, "Steganography and Steganalysis in Digital Images," Polytechnic University of Puerto Rico, San Juan, 2012



List Of Segments

Secrit Message Size	32 Bytes. Want 64 Pixel
Cover Image Size	18 X 18

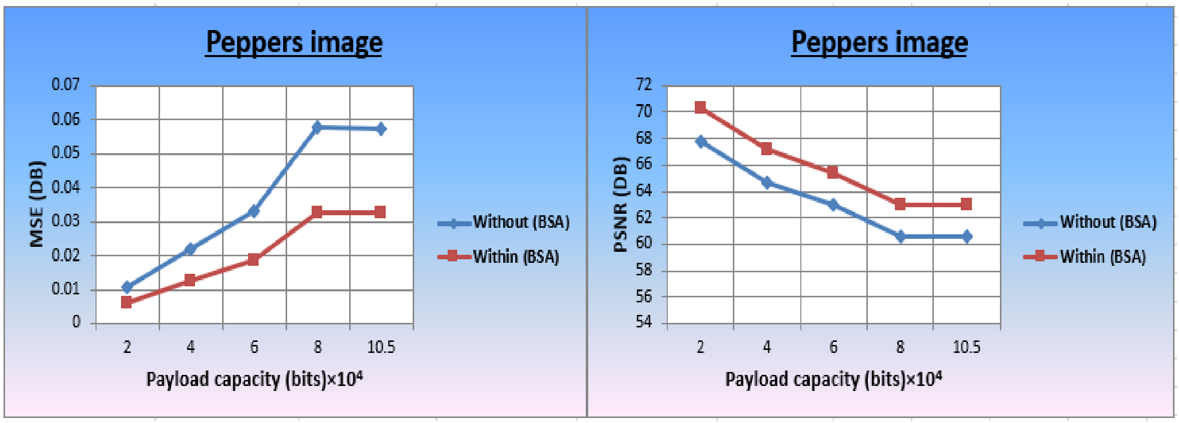
XReal	YReal	KeyX	KeyY	SegX	SegY	PerPx	PerPy	PerV
4	16	1	1	4	16	2	1	113
4	16	2	1	8	16	0	1	110
4	16	3	1	12	16	0	0	114
4	16	4	1	16	16	0	0	124
8	8	1	1	8	8	8	7	112
8	8	1	2	8	16	5	1	104
8	8	2	1	16	8	1	3	113
8	8	2	2	16	16	1	1	115
16	4	1	1	16	4	1	11	112
16	4	1	2	16	8	1	7	114
16	4	1	3	16	12	0	0	112
16	4	1	4	16	16	0	0	110

Figure 5. Segmentation method

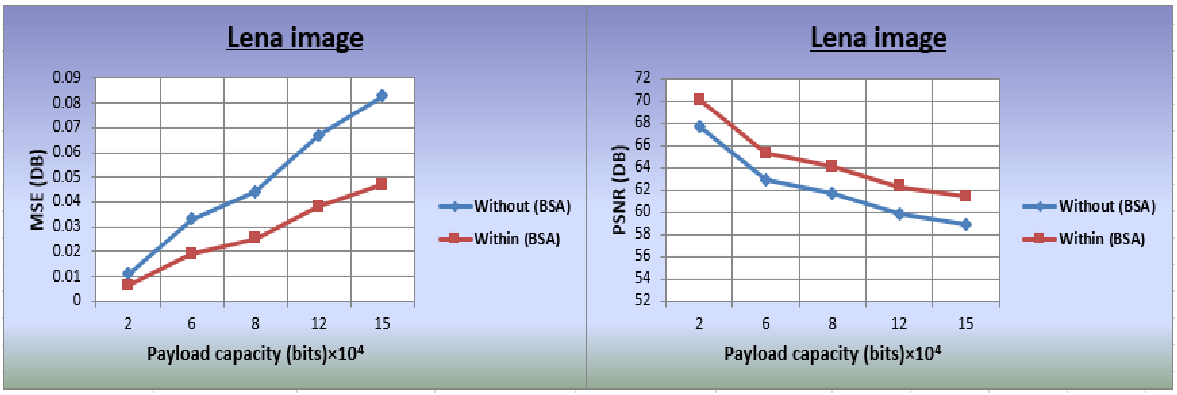
Table 5. Performance comparison between the proposed approach and other proposed techniques

Images 512x512	Payload capacity (bits)×10⁴	PSNR(DB) Using Elshare, S., EL-Emam, N (2018)	PSNR(DB) Using Ali, M, A., (2019)	PSNR(DB)using the proposed algorithm
	2	61.2	62.6	67.7
	4	59.1	60.9	64.7
	6	55.3	56.6	62.9
	8	53.7	55.6	60.5
peppers	10.5	53.7	54.3	60.5
	2	62.7	63.2	67.7
	6	57.3	58.1	62.9
	8	55.6	56.3	61.6
	12	54.8	54.9	59.8
Lena	15	53.1	53.7	58.9
	2	59.2	60.4	67.8
	2.8	56.1	57.9	66.3
	3.6	54.9	55.1	65.2
	4.4	53.2	53.4	64.2
Baboon	5.6	53.4	53.7	63.1
	2	63.2	65.1	67.7
	5	59.6	62.4	63.7
	7	58.5	59.7	61.6
	10	55.9	56.2	60.7
Barbara	12.5	54.8	55.1	59.7

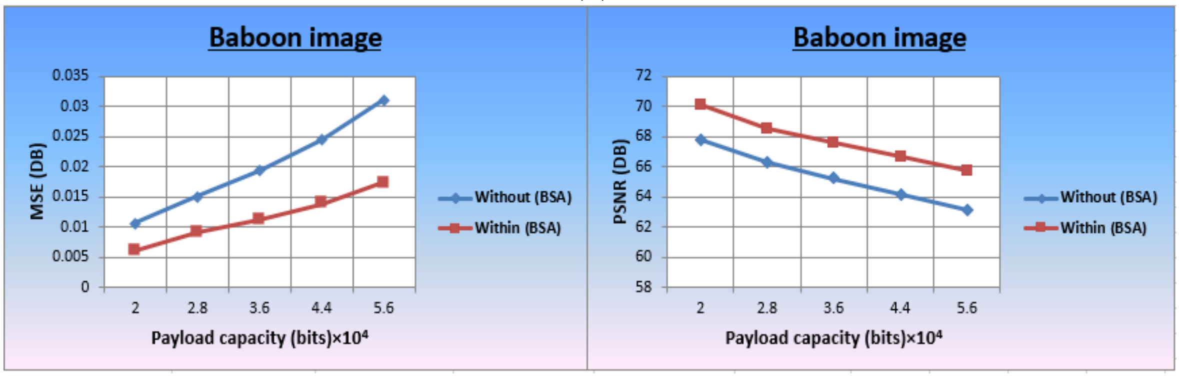
(A)



(B)



(C)



(D)

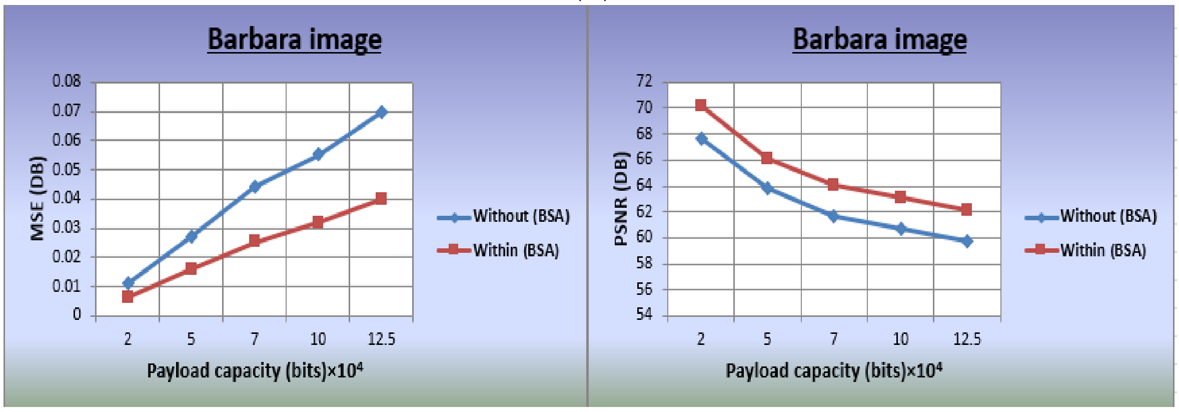


Figure 6. Comparison between MDHEA with BSA and without BSA as : (**A**) Comparison of distortion between MDHEA with and without (BSA) for peppers image; (**B**) Comparison of distortion between MDHEA with and without (BSA) for Lena image; (**C**) Comparison of distortion between MDHEA with and without (BSA) for Baboon image; (**D**) Comparison of distortion between MDHEA with and without (BSA) for Barbara image