Computer Systems Science & Engineering
DOI: 10.32604/csse.2022.023904
Article

Tech Science Press

An Animated GIF Steganography Using Variable Block Partition Scheme

Maram Abdullah M. Alyahya[1], Arshiya S. Ansari[1,*] and Mohammad Sajid Mohammadi[2]

[1]Department of Information Technology, College of Computer and Information Sciences, Majmaah University, Al-Majmaah, 11952, Saudi Arabia
[2]Department of Information Technology, College of Computer, Qassim University, Buraydah, 51452, Saudi Arabia
*Corresponding Author: Arshiya S. Ansari. Email: ar.ansari@mu.edu.sa
Received: 26 September 2021; Accepted: 16 November 2021

Abstract: The paper presents a novel Graphics Interchange Format (GIF) Steganography system. The algorithm uses an animated (GIF) file format video to apply on, a secured and variable image partition scheme for data embedding. The secret data could be any character text, any image, an audio file, or a video file; that is converted in the form of bits. The proposed method uses a variable partition scheme structure for data embedding in the (GIF) file format video. The algorithm estimates the capacity of the cover (GIF) image frames to embed data bits. Our method built variable partition blocks in an empty frame separately and incorporate it with randomly selected (GIF) frames. This way the (GIF) frame is divided into variable block same as in the empty frame. Then algorithm embeds secret data on appropriate pixel of the (GIF) frame. Each selected partition block for data embedding, can store a different number of data bits based on block size. Intruders could never come to know exact position of the secrete data in this stego frame. All the (GIF) frames are rebuild to make animated stego (GIF) video. The performance of the proposed (GIF) algorithm has experimented and evaluated based on different input parameters, like Mean Square Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) values. The results are compared with some existing methods and found that our method has promising results.

Keywords: (GIF) Steganography; frame partition; variable data insertion; data encapsulation

1 Introduction

The primary objective of Steganography is to hide not only confidential information but also the presence of confidential data in the cover media instead of encrypting it [1]. Information security is a very important issue over the Internet and Steganography skills are very useful for the future of Internet security and privacy for secrete communication. Using steganography methods, a good amount of variety of data one can hide and store in various cover mediums to avoid information leakage. Fig. 1 shows the basic steps of steganography. Steganography allows smooth secret data transmission and helps to reduce issues of information security and authorization which has become a critical factor now. The steganography can be categorized as fragile steganography which is easy to lookup for the embedded

message and robust steganography which need much effort to try to look up for embedded message [2]. The Steganography system is very valuable for confidential data transfer applications like; to transfer secret text, image, audio, or video from its source to the desired destination, to store and transmit confidential location information. Other applications could be Secure online voting, Private banking, Military purpose, Software Company, Film Industry, and many more.

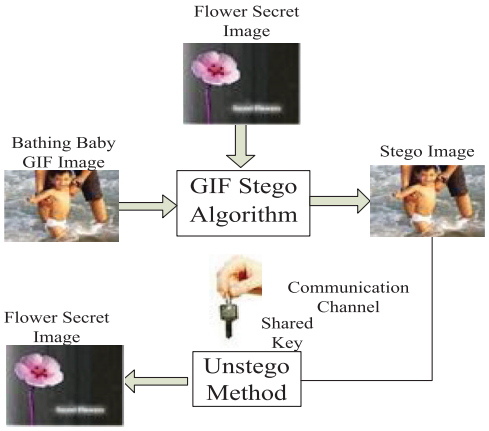

Figure 1: (GIF) Steganography steps

The purpose of this work is to provide improvement in security of steganography on animated images specially on (GIF) format. Our (PSNR) and Steganalysis results should prove the better quality and visualization of (GIF) stego images.

2 Literature Review

In the field of information security, Steganography is an important topic of discussion. By Showing the importance of (GIF) [3] firstly, here we have reviewed the work related to Steganography published by some authors [3–19]. and the important point from these papers is extracted and presented in Tab. 1.

Table 1: Literature review

Year	Author & reference	Methodology used	Database used & results
2013	Bakhshi et al. [3]	The authors have used a Tumblr. It is a big social networking micro blogging platform where users store animated (GIF) by blobs or posts by like or reblog etc.	Used 100 k animated (GIF) after interchanged with user they found that what makes animated (GIF) engaging its low bandwidth, minimum time demands and utility of expression of emotions are significant factors to be considered for engagement in Tumblr.
2010	Tiwari et al. [4]	Here author presented (GIF) image steganography using (LSB) method. The message has been hidden in the (LSB) of the pixels of (GIF) images.	Internet based (GIF) images have used for experiment. Results of Stego1Bit insertion is considered robust whereas Stego2Bits, Stego3Bits, and Stego4Bits examined decrease in perceptibility.

(Continued)

Table 1 (continued)

Year	Author & reference	Methodology used	Database used & results
2013	Abed Elgabar et al. [5]	Here in this paper author presented comparison of (LSB) and (GIF) Steganography methods.	Concluded Bitmap (BMP) image format is good for (LSB) method as compared to (GIF) file format but more prone to attack. Different (BMP) and (GIF) images from different papers studied here. Authors proved that (GIF) images contain little data and have low resistance to attacks. The image gets messed up, distorted and subject to detection as the amount of embedded data increases.
2013	Abed Elgabar et al. [6]	Compared and analyzed the Least Significant Bit (LSB) algorithm using the cover object as an image in (JPEG) and (GIF) format to understand the strengths and weaknesses of each format.	According to the author's result of (GIF) contain very little data. More data insertion bits can easily identify the modification.
2015	Abed Elgabar [7]	Hide secret image inside cover image using (LSB), so no one will discern the hidden image in cover file. Two types of image formats used as the cover were compared (GIF), (JPEG).	(LSB) in (JPEG) is better than used in (GIF) because the size of the embedding data in (GIF) image format is lower than in (JPEG). Also, it was observed that when the embedding data in (GIF) format were increased, this would distort the cover image.
2016	Munir [8]	Hide secret messages in animated (GIF) by applying modified EzStego algorithm. So that the parts of the message are randomly included in each frame.	The result for stego image is better and there is no significant deterioration found in quality measurement.
2016	Juzar et al. [9]	The method for embedding messages in an animated (GIF) implemented using multibit assignment.	Here revealed if method is fully depends on colors of (GIF) image then they are more prone to attack.
2017	Miltner et al. [10]	Textual and visual analysis used to examine the content of (GIF) files and to discuss (GIF)'s features, capabilities, and broader relevance to digital culture and communication.	Here authors suggested to use the (GIF) cover . The reasons to prefer the use of (GIF) format is because it displays images with endless frequency and allows them to send multiple images in one (GIF).
2017	Fathurohman et al. [11]	Steganography technique using Least Significant Bit (LSB) and Adaptive method applied to (GIF) images.	Here Youtube (GIF) images and character data have taken as an input. The robustness of the (GIF) was different, concluded adaptive method is better over the (LSB) method.
2020	Abood [12]	Used a new steganography method to hide data securely in a dynamic (GIF) frames. Used Syndrome-trellis codes (STC) framework algorithms for cost and load allocation for different frames.	The algorithm achieved higher security compared to previous work and the experiments were performed on a database containing 500 dynamic GIFs.
2017	Amirulhaqi et al. [13]	Explain the importance of security using Steganography and a comparison between Spread Spectrum and (LSB) in (GIF) images.	Spread spectrum method achieves more security compared to (LSB). Stego perceptibility is closer to 50 (dB) decibel. If the

(Continued)

Table 1 (continued)

Year	Author & reference	Methodology used	Database used & results
			PSNR > 50 (dB), it consider an ideal performance.
2018	Basak et al. [14]	Authors have used Modified Pixel-Value-Differencing (PVD) method and hash function to embed message into (GIF) file format.	Here grey scale animated (GIF) is used for experiment results showing capacity around 74KB and on an average (PSNR) range 36 to 40 dB on 74KB secret data.
2018	Hashim et al. [15]	Hiding image as it is the most common branch of steganography. Further detail is shown based on the (LSB) within different image formats.	According to the authors, to evaluate the performance of the new scheme criteria must be used, namely, hiding capacity, safety, and distortion management.
2020	Lin et al. [16]	A new steganography method to hide security data in a dynamic (GIF) when using the (STC) framework, including algorithms for cost and load allocation for different frames.	The algorithm achieved higher security compared to previous work and the experiments were performed on a database containing 500 dynamic (GIF).
2021	Zhu et al. [17]	The reference images were generated then modified, adaptively embedding the pixels and performing +1 and −1 operations in the (RGB) color space.	Experimental results demonstrated the superiority of security performance over the method of steganography for animated emoji images.
2020	Gupta [18]	Steganography method that converts a color palette and divides it into (RGB) cubes with only one color. This method use in two formats only, (GIF) & (PNG).	The comparison between (GIF) and (PNG) has been observed that the (PNG) image is best suited to the (LSB) format, when the focus is on the amount of information sent, whereas one of its disadvantages is that not all web browsers can support (PNG).
2020	Mstafa et al. [19]	Here offered a safe and invisible way to Steganography and embed sensitive data and information in digital video	Method is better than other previous methods, and it turned out to be better in terms of visual perception, results shown (PSNR) around 60.7 dB, and excellent for frames reunion ability.
2021	Basak et al. [20]	Here presented Steganography method based on (LSB) replacing (ASCII) codes to hide a secret message in an animated frame. The secret text also get encrypted using (SHA1) to improve security.	The algorithm has been tested on several (GIF) color images, and the results are good with the ability to include high data and maintain the visual perception of the output images.
2021	Lubis et al. [21]	make a modification of the (LSB) with insertion based on the length of the message in (JPEG), (BMP) and (GIF).	The result for Stego image is better, in text message. (PSNR) range is 66.29 decibel, in image messages (PSNR) range is 54.20 decibel.
2020	Ansari et al. [22]	Here introduced new Generic steganography algorithm (GSA) for hiding data in multiple, different formats images. Concepts such as adaptive segmentation systems with data	Experimental results shown for different used images are around (PSNR) 59%.

(Continued)

Table 1 (continued)

Year	Author & reference	Methodology used	Database used & results
		propagation were used to embed confidential data.	
2020	Chen et al. [23]	If extraction of image performed using convolutional network method there is much possibility to loss the image quality. Here in this research authors have presented segmentation fusion model using layer by layer method to optimize the image 6.3 percent as compare to convolutional method.	They have used PASCAL VOC 2012 and PASCAL Context datasets for experimentation.
2020	Luo et al. [24]	This paper has proposed the deep learning, (CNN) and (DCT) based method for hiding data in real time image. This method hide data without modifying the pixels of image in coverless real time images.	Here they have used online real time images to test results.

According to the published work, this literature review explains how the bits get inserted into different file formats and what are their embedding zone as shown in Fig. 2. It also explain different methods used by different authors to embed data in their algorithms.

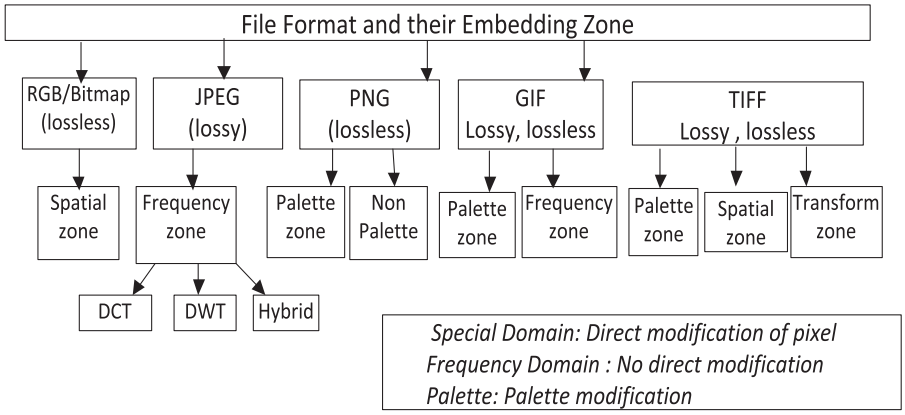

Figure 2: Steganography secret data embedding zone

Basically, there are three kinds of domains as shown in Fig. 2 where the data bits can be inserted in. First one is Spatial Domain (RGB/Bitmap) Steganography, second is Frequency Domain Joint Photographic Expert Group (JPEG) Steganography and third is Palette base or Image data base (PNG) Steganography. Here we have discussed the number of Steganography methods to insert data like Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), Distortion Method, Image Statistical Method, Image Adaptive Method, very common Least Significant Bit Substitution, strong Indicator technique method, and Palette modification method.

The reviewed papers reveal following 3 important conclusions about embedding domain:

1. Spatial Domain offers big amount of capacity but less safety. It is because image pixel can be altered directly as per the picture's curves, image colors images borders or edges.
2. The Frequency Domain method is more vulnerable than spatial domain method.
3. (GIF) and (PNG) Palettes based method is extremely secure but not be able to provide more capacity. (GIF) and (PNG) image base method of embedding gives better capacity.

Using one method can we insert data in all formats? The answer is no. we cannot insert data in all image format using one method because all image format having different characteristics. Like in Bitmap image format we can insert data directly by modifying pixels whereas in (JPEG) for there are coefficients which cannot be modify directly. Even small modification in (JPEG) format shows great destruction in (JPEG) image. Similarly, (PNG) image format is pallet base where we have to change pallet colors to hide data bits [25].

In this work, we have used (GIF) frequency-domain based way to embed data on (GIF) color animated frames. To understand the (GIF) data insertion procedure. It may be important to have a look and understand (GIF) File Format details. As shown in Fig. 3 the (GIF) file format has separate headers, descriptors, image data, file trailer blocks etc. to store all kind of image information in them.

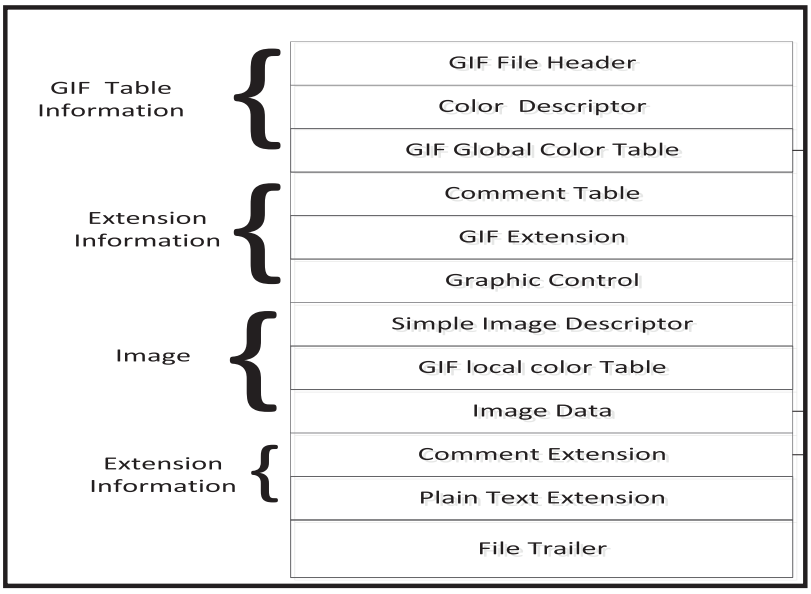

Figure 3: (GIF) file format

It is originally an image file format that is animated by combining several images into one file to display them in sequence to create an animation clip or short video. It is a format that uses an indexed color scheme, and single (GIF) image palette is limited to 256 color and supports up to 8 bits per pixel per image, which makes optimizing the visual image weight possible this means it is the ideal format for images which use only specific color values and is not suitable for images color variety such as digital photography. (GIF) format is best suited for graphics, cartoons, logos, graphics, as well as simple animation [3]. (GIF) is the only option for placing animations online without the need of flash. (GIF) images are compressed with lossless compression, and the file size is very small. The version GIF89a support for delayed animation, storing of metadata and transparent backgrounds to make the multi-image storage feature more useful for animation.

(GIF) format is now very popular because they are often used as emotional reactions in blogging, social media, and instant messaging apps. Most steganography tools rely on (JPEG), (BMP), (TIFF) and other formats, but many surveys and research have not devoted on (GIF) animation, due to the belief that embedded messages may affect the (GIF) cover more than regular images, because the color palette is weak which includes Only 256 Red, Green and Blue (RGB) colors.

3 Material & Methods

This section reports proposed (GIF) Steganography algorithm. Fig. 4 shows the detail explanation of our (GIF) steganography method with the help of a block diagram.

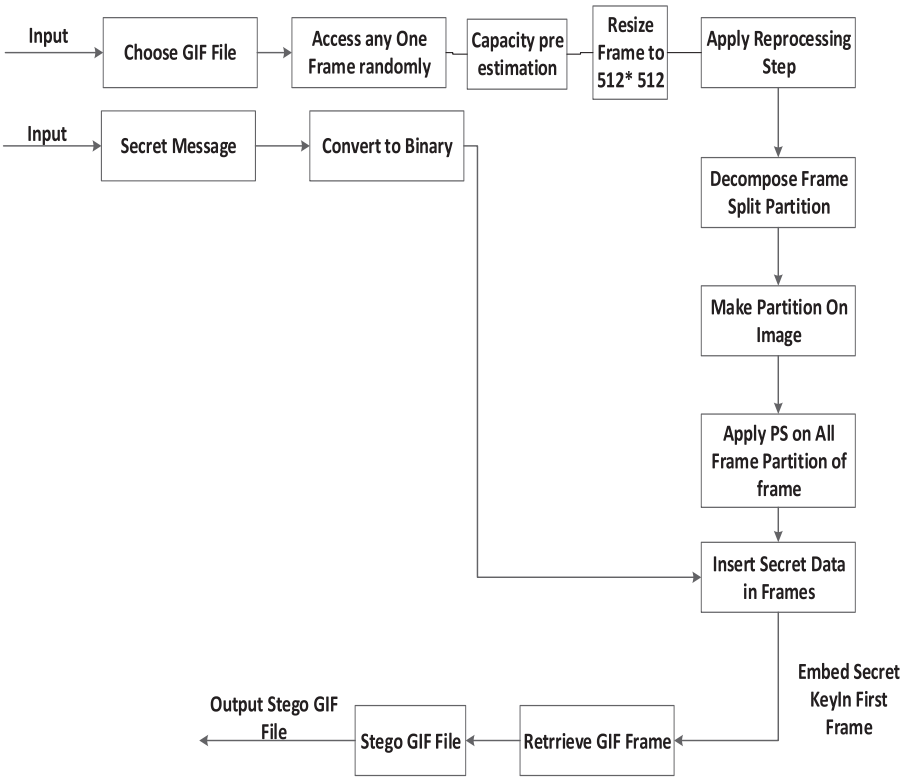

Figure 4: Block diagram of proposed (GIF) steganography process

Firstly, our algorithm scans a (GIF) cover object and conceal data (say text paragraph or an image) which is to be hide in it, after that, it picks out one of the (GIF) frame from the number of frames of animated (GIF) cover file. At that instant capacity estimation is done. After that it applies different types of conversions, like color to grayscale, resizing and filtering to smooth the image using Gaussian filter. The method uses a block partition method that need to coordinate with four different portions of the selected frame. Minimum 4 dimensions' image decomposition is used here so we get variable partitions of 512, 256, 64, 32, 16, 8, 4, 2 and 1 (as we are resizing frame 512×512). Thus, based on it we are getting their variable partition and the final partition value is 1. It creates variable size image block partition structure in the separate frame as show in Fig. 5.

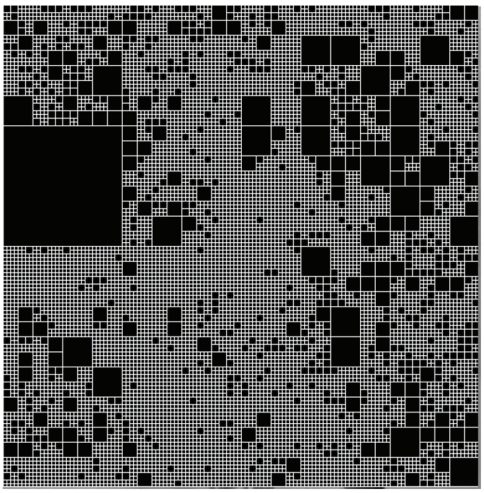

Figure 5: Image block partition structure

For block partition we are using Quad-tree decomposition method. It is a technique that divides an image into various sub blocks which are more homogeneous compared to the original image. This decomposition process provides all image related information including the structure of an image. This technique is widely used in various applications such as texture analysis, image compression etc. The Quad-tree decomposition can be achieved with the help of the auto qtdecomp function. This auto function works by performing image segmentation. In this function a square image is divided into four equal-sized square blocks. The divided square block check whether they satisfy the homogeneity criteria, it is validated to see whether all the pixels in the image blocks lie within a predefined dynamic limit. All blocks satisfying the criteria are not subjected for further division. The blocks which do not meet the criteria are subdivided again into four blocks and are tested again using the testing criteria. This iterative process is continued until all blocks of the original image meets the specified testing criteria. The outcome of the process might result in the creation of blocks with different sizes. The size of a block can be as low as 1×1 unless a specific size is defined. The function qtdecomp returns the quadtree decomposition in the form of a sparse matrix whose size is the same as that of the original matrix I. The nonzero elements in the matrix constitute the upper left corners of the blocks and the size of each block is determined by obtaining the value of each nonzero element in the matrix.

Secret data get embedded on selected frame after incorporating the block partition structure with selected (GIF) frame that generate mixed frame with frame data and image block partition structure. The algorithm uses a shared key to select frame randomly for data embedding thus impostor could never know in which frame the data is embedded. The Shared key is also stored as seed to generate same random number sequence at receiver side. It is stored in the first frame (sender and receiver already agreed upon this protocol) so that decoder would able to know exactly in which frame the data is embedded with the help of same generated random number. After embedding the data, it needs to extract the image from this mixed file i.e., so it needs to separate the image block partition structure from mixed frame to hide it as shown in Fig. 6 (frame 6 and 7). The algorithm again merges all the frame including above processed stego frame and again converting it to animated (GIF). Animated stego (GIF) file as shown in Fig. 4. Precisely opposite or reverse procedure can apply to get back data frame at receiver side. Through algorithm we describe the various other steps of the proposed method in further details as follows.

The Algorithm for proposed method is as given below.

Pseudo Algorithm: (GIF) Steganography.

Input: (GIF) Cover Image, Secrete Data, Shared key for data extraction.

Output: (GIF) Steganography Image.

Begin

1. Take any Animated (GIF) video and extract (GIF) frames out of it.
2. Resize the frames to 512 × 512 normal size as option of input cover image.
3. Separate and select anyone (GIF) frame randomly to hide data using a seed.
4. Convert from color to grey scale.
5. Apply Gaussian filter.
6. Generate a partition structure and apply it on selected frame image.
7. Decompose frame split into variable partition 512, 256, 64, 32, 16, 8, 4, 2, 1
8. Embed variable amount of data bits in these variable partitions.
9. Embed shared key as seed in first frame.
10. Extract partition block structure from this frame.
11. Reconstruct Animated (GIF) from all frames.

End

The algorithm uses variable partition blocks for data embedding in (GIF) file after taking, resizing, color conversion and filtering the image and its frame. Decomposing the (GIF) selected frame into variable blocks then putting data bit in those data blocks. Variable size of data will be embedded in those blocks as block itself is of variable size.

Let us take an example of data embedding algorithm used to insert data and select indicator key pixel using pixel x, y, z. As shown in Tab. 2 the pixel x, y, z selected randomly in circular order x → y → z→ x. Here in Tab. 2, we select pixel z to denote indicator pixel key through the random selection. This pixel will not store any data bits. Secret data will store in lowest valued pixel of remaining two pixels. Here in example 'y' value is having lowest value so the secret data will store in y. If after modification of 'y', the 'y' value will become greater then 'x' value then we will modify LSB of 'x'. It is done here to be able to know which pixel was the smallest 'y' or 'x' while we extraction data in data extraction process form stego file.

Table 2: Indicator method and storing data in the (GIF) pixel

Step	Action	Pixel 'x'	Pixel 'y'	Pixel 'z'
0	Pick up the frames from number of available frames of animated (GIF) & selected 3 pixels x, y & z for first selected partition block.	34	30	25
1	Pixel 'z' is indicator and is 'ignored'.	34	30	25
2	Smaller pixel value 'y' is selected.	34	30	25
3	Convert dec values to bin.	00100010	00011110	25
4	Here embed secret bit data (1 bit as value = 1).	00100010	00011111	25
5	Equivalent decimal.	34	31	25
6	Not modification in (LSB) of 'x' in this case.	00100010	0011111	25

4 Results

The proposed algorithm is experimented on number of (GIF) images. All cover (GIF) images are resized to 512 × 512 images as used in Figs. 6–9 for explanation simplicity and illustration purpose, we discuss only Bird and Duck cover (GIF) files.

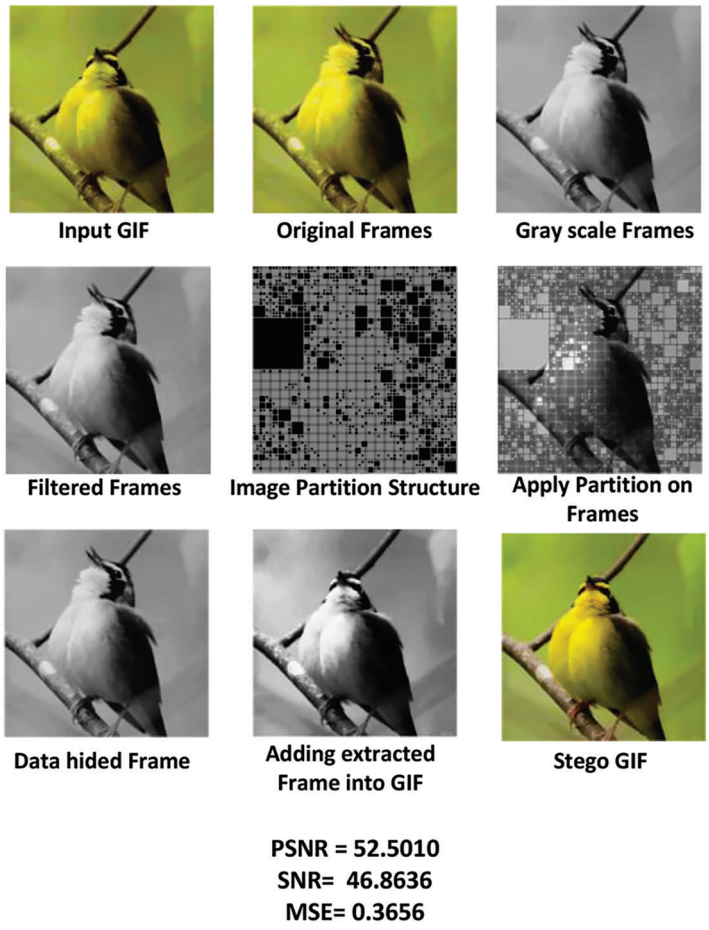

Figure 6: Implementation of (GIF) steganography steps with result on BIRD (GIF)

Fig. 6 shows output stego images of Bird obtained on different partition blocks (p_i) where i = 1 to 9 like ($p_1 = 512, p_2 = 256, p_3 = 64, p_4 = 32, p_5 = 16, p_6 = 8, p_7 = 4, p_8 = 2, p_9 = 1.$) The equivalent PSNR and MSE results are shown in Tab. 3 and Fig. 6 respectively.

The performance measure PSNR is given by Eq. (1) as below [22]:

$$PSNR\ value = 10 \times \log 10\left(\frac{255 \times 255}{MSE}\right)(\text{dB}). \tag{1}$$

where, the MSE (Mean Square Error) is given by:

$$MSE = \frac{1}{(m \times n)} \sum_{i=1}^{m} \sum_{j=1}^{n} (gc_{ij} - p_{ij})^2 \tag{2}$$

Here, gc_{ij} is (GIF) cover image and p_{ij} is (GIF) stego image coordinate's pixel values.

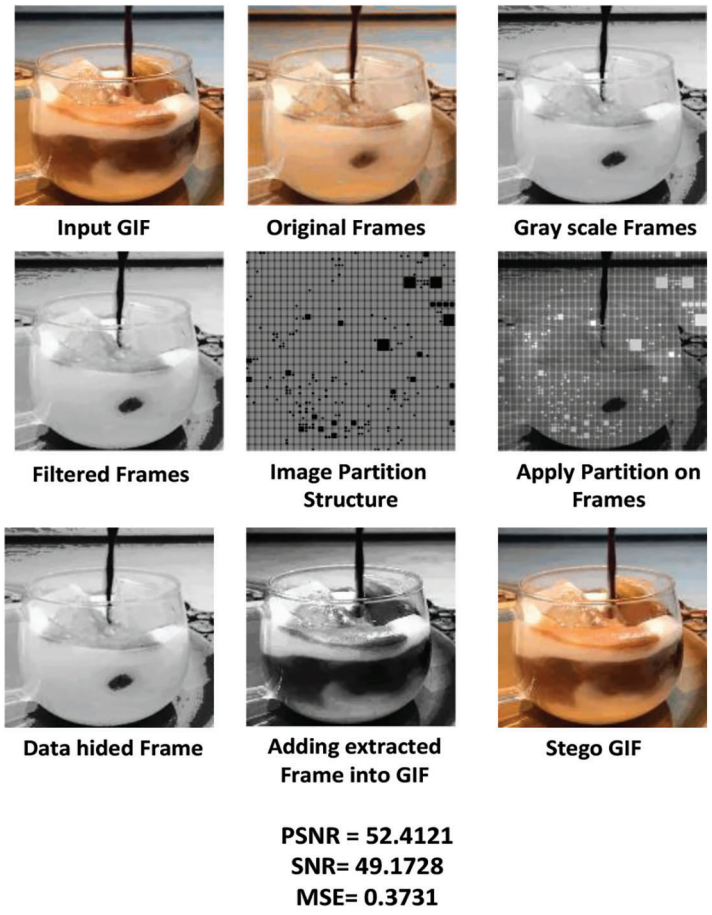

Figure 7: Implementation of (GIF) steganography steps with result on GLASS CANDLE

The (PSNR) calculating the square root of the quantity of photons in the brightest portion of the image. This is a very positive feature producing several photons ensuing little noise, thus more similarity could be observed between cover file and stego file.

5 Discussions

5.1 Imperceptibility Evaluation

The (GIF) Steganography imperceptibility assessment test is based on Secret Data Inserting Rate & Data Capacity Utilization of (GIF) cover image.

Let GIF_HDB be the actual number of hidden data bits in the (GIF) cover image and let (SDIR) be the Secret Data Inserting Rate. Then,

$$(\ SDIR) = \left(\frac{GIF\_HDB}{GIF\_CCapmax}\right) \times 100 \text{ Bits} \tag{3}$$

Let GIF_CCapmax be the maximum (GIF) Cover Capacity of bits, for particular (GIF) image. The GIF_CCapmax is given by,

$$GIF_{CCapmax} = \left(\text{Total number of blocks } \times \frac{avg_i \times \text{avg}_{j \times 3}}{2}\right) \times \ bits\ per\ pixel \tag{4}$$

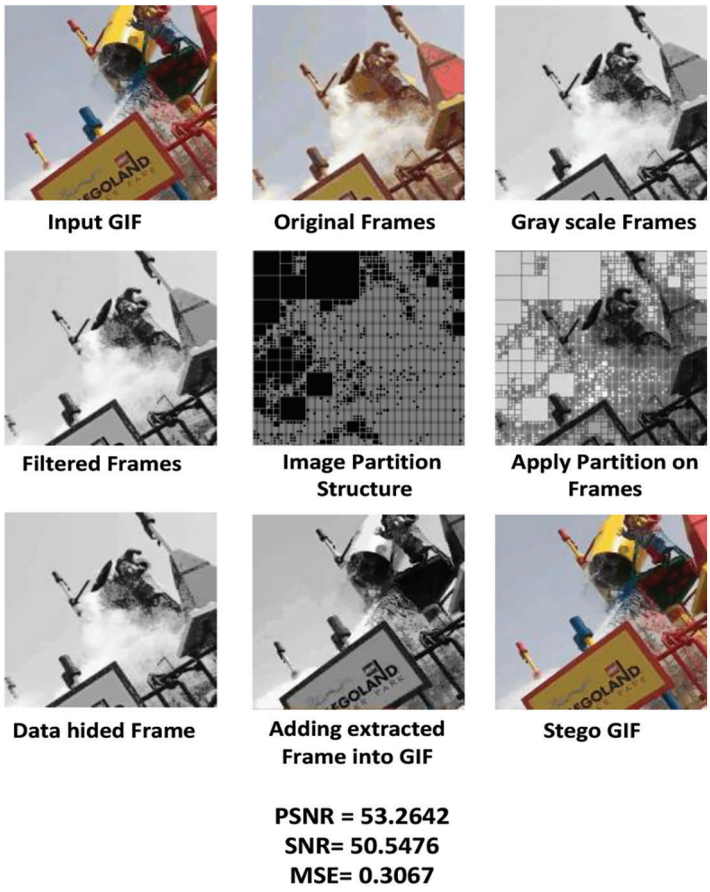

Figure 8: Implementation of (GIF) steganography steps with result on TEMPLE (GIF)

where, for the Number of bits could be as minimum as one (can be increased) bit per pixel so it has value 1 and it is preferred by conducting tests on different (GIF) images that retain the stego images undistorted. Variables *avg_i* and avg_ j are the average number of rows and columns of total blocks respectively. Constant values 3/2 for the covering all the three image plane pixels and due to second and third image plane has only quarter of total pixels in each of them. The value for GIF_CCapmax is approximately 98 KB bits for (GIF) steganography algorithm for a standard cover image size of $(512 \times 512 \times 3)$ while retains its stego image perceptibility acceptable.

Tab. 3 shows imperceptibility test results based on different secrete data insertion rate and different GIF_CCapmax utilized (25%, 50%, 75%, and 100%) of (GIF) cover images. Under GIF_CCapmax capacity range, the secret bits are embedded in $(512 \times 512 * 3)$ sized images frames, using different embedding rates of 100% to 25%, different secret data approximately equal to 97KB, 68KB, 45KB and 22KB are used to utilize embedding capacity as per the need. The Measuring parameters (MSE) and (PSNR) prove imperceptibility test of the algorithm. Figs. 6–11 shows the output result of our algorithm. If we discuss the stego image for 'Bird' and 'Temple', at different Secrete Data Insertion Rate (SDIR) and different (GIF_CCapmax) values, observation shows that at 100% Embedding Rate, we could hide 90,000 bits; even at 100% capacity utilization, (PSNR) shows promising output. We can observe in Tab. 3 that (PSNR) is inversely proportional to Secret Data Inserting Rate, Embedding Capacity Utilization, and Total Number of Secret data bits. Thus, decreasing (MSE) and increasing (PSNR) for all cover images is observed. This observation gives rise to achieved better imperceptibility. The (MSE),

(PSNR) readings of Bird and Temple cover images are approximately same with little difference, because our algorithm uses the same sized Cover image and fix number of bit insertion per pixel.

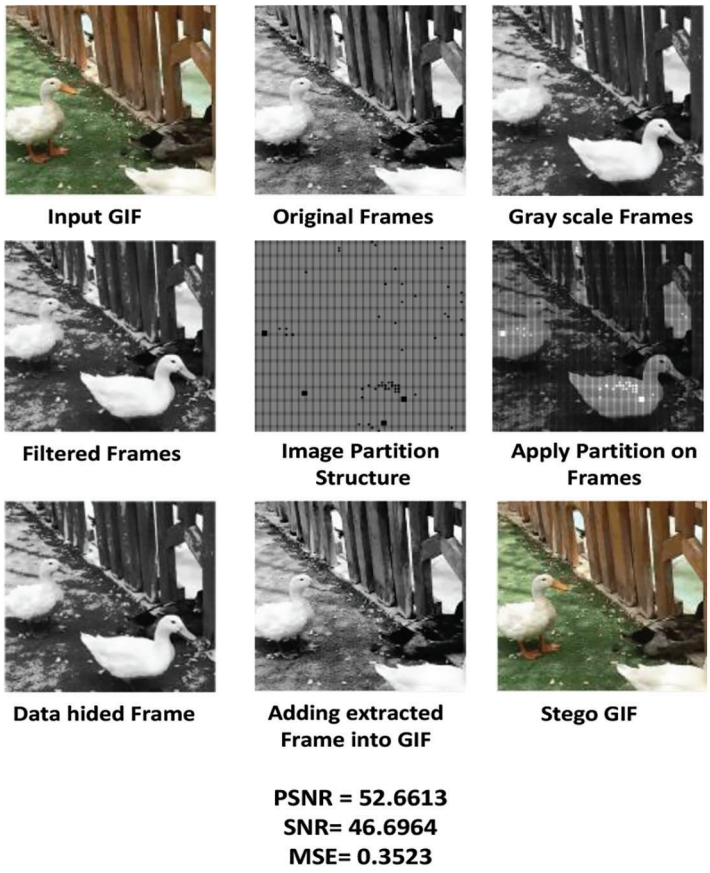

Figure 9: Implementation of (GIF) steganography steps in result on DUCK (GIF)

Table 3: PSNR values based on percentage embedding rate and capacity utilization of cover image

(GIF) Secrete data insertion rate $\%SDIR$	100% capacity utilized, $SDIR = \sim 100\%$ at 90KB		75% capacity utilized, $SDIR = \sim 75\%$ at 68KB		50% capacity utilized, $SDIR = \sim 50\%$ at 45KB		25% capacity utilized, $SDIR = \sim 25\%$ at 22KB	
Test Data Size (512 × 512)	PSNR (dB)	MSE	PSNR (dB)	MSE	PSNR (dB)	MSE	PSNR (dB)	MSE
(GIF) BIRD	52.5010	0.36	55.2110	0.24	57.9810	0.21	58.9810	0.98
(GIF) TEMPLE	53.2642	0.30	55.1801	0.24	57.5612	0.21	58.5612	0.97
(GIF) GLASS CANDLE	52.4121	0.37	56.0810	0.20	58.5671	0.19	59.5671	0.94
(GIF) DUCK	52.6613	0.35	56.8231	0.20	58.2100	0.19	59.2100	0.92

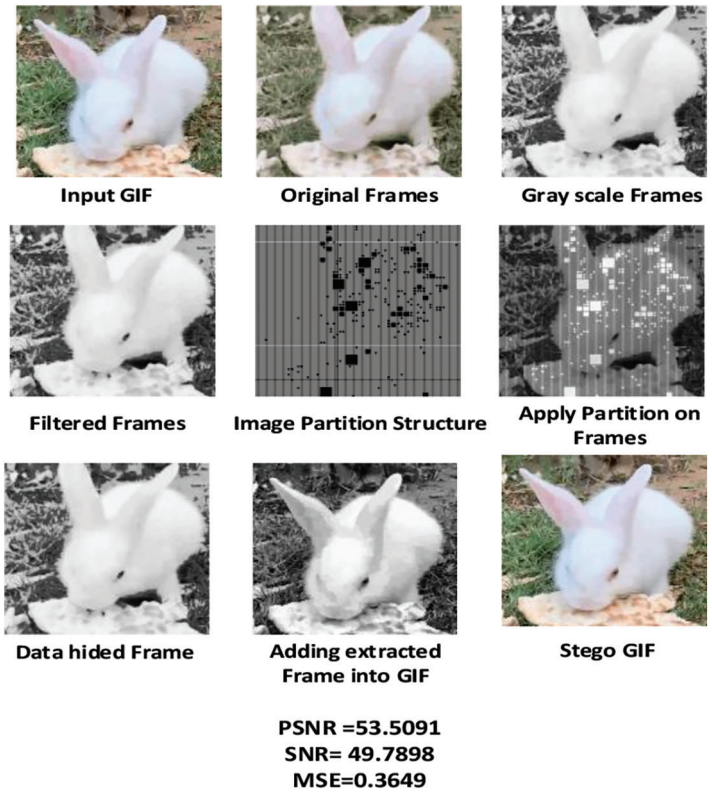

Figure 10: Implementation of (GIF) steganography steps with result on RABIT (GIF)

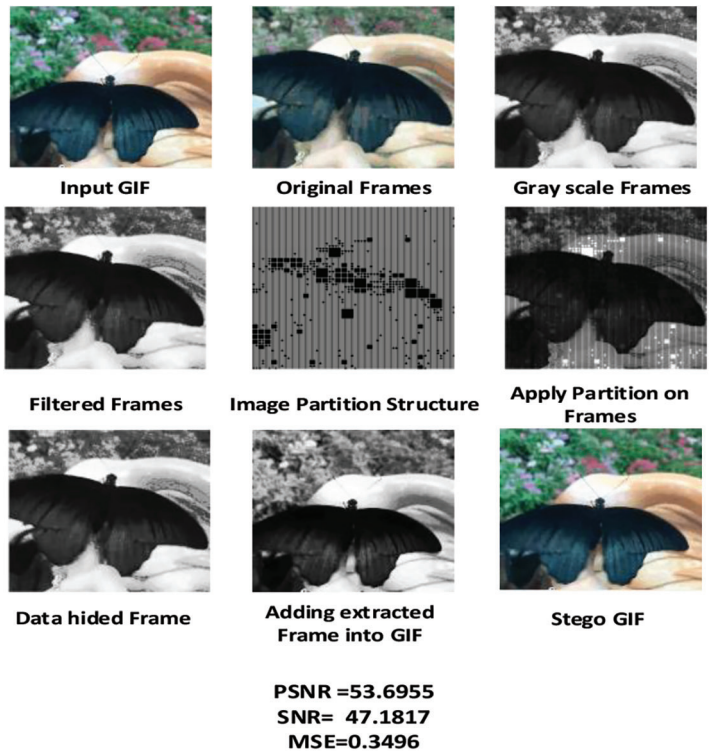

Figure 11: Implementation of (GIF) steganography steps in result on BUTTERFLY (GIF)

5.2 Result Comparison with Other Works

Proposed (GIF) steganography algorithm results are compared with methods in [14,20] and shown in Tab. 4. These results tested on images of same dimension 512 × 512 and compared based on equal capacity. Methods [14] have shown results using data around 70 kilo bytes, whereas we have used around one 93 kilo bytes secret data.

Table 4: (PSNR) comparisons of the proposed algorithm with other (GIF) Steganography methods

Cover Image Size (512 × 512) Our paper image Vs All images used in other papers	**Secret Data Capacity in KB**	**Proposed Scheme PSNR in dB**	Secret Data Capacity in KB	Pixel value differencing Method [14] PSNR in dB	Secret Data Capacity in KB	(LSB) _GIF Method [20 SGSAHP Method] PSNR in Db	Secret Data Capacity in KB	ASCII substitution method [20] PSNR in dB
TEMPLE.GIF	**92KB**	**53.08**	*72.46 KB*	*33.76*	47 KB	35.92	*90 KB*	*40.1*
DUCK.GIF	**90KB**	**52.61**	*73.46 KB*	*36.57*	47 KB	31.90	*90 KB*	*37.2*
GLASS CANDLE.GIF	**90KB**	**52.41**	*93.46 KB*	*36.59*	47 KB	34.21	*90 KB*	*42.09*
BIRD.GIF	**90KB**	**52.50**	*80.27 KB*	*32.22*	47 KB	31.11	*90 KB*	*42.86*

The (PSNR) Value of our algorithm on even maximum capacity utilization is higher than the (PSNR) of the other two methods given in [20]. Also that the proposed (GIF) stego algorithm has high (PSNR) almost the same values for all cover images. In contrast, other method in [14] has low value for its capacity and (PSNR) values even for less amount of secrete data.

5.3 Robustness Test

Steganalysis is the art of breakage the Steganography method. It is also one of the procedures to test the robustness of Steganography [21]. Researchers have presented multiple Steganalysis approaches in [26,27].

A secret object can be detected or separated from the cover file by the Steganalysis tool as shown in Fig. 12. We have used "Ben4D" Steganalysis tool to check the strength of our algorithm. We have also done a security check for our method using this Steganalysis "Ben4D" tool and it proves the robustness of (GIF) steganography algorithm. We also have tesed our stego image on other steganalysis tool "Try it Out" and found that no detection is reported by the tool as show in Fig. 13. Steganalysis detects the stego image when it finds uneven or heavily loaded area in image. To reduce the effect of modification we have balanced the unmodified bits by inserting alpha bits into unmodified bits to balance the weight of the pixel.

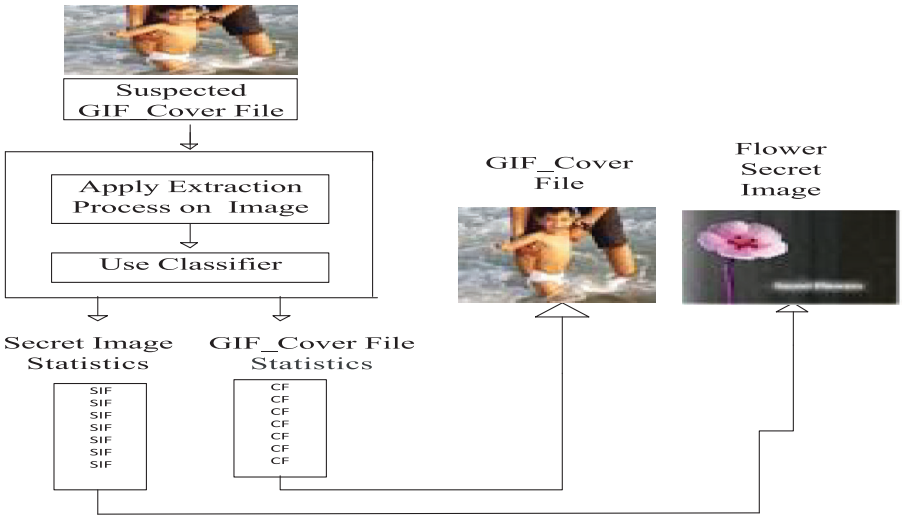

Figure 12: General Steganalysis process

Figure 13: Test of Steganalaysis

This procedure does not distinguish the difference between modified and unmodified pixel and hence Steganalysis would not be able to detect the stego out image. Tab. 5 shows the Ben4D and Try it out steganalysis tested results.

Table 5: Steganalysis test result

File type tested	Tool used	No. of images tested	Sample match
GIF	BEN4D	10	No editor does not appear in list
GIF	Try it out	10	No trace of data embedding found

6 Conclusion and Future Work

This paper presents a novel animated (GIF) Steganography work to enhance the security concerns of data transmission over the Internet. It gives good quality stego output file and provides the optimum imperceptibility. The fundamental element of the algorithm is indicator key, data embedding algorithm and variable partition blocks to increase robustness on even higher capacity. Partition Scheme could choose the random partition block and random frame selection in this way better security can achieved by inconsecutive scattering of secret data in every block of selected (GIF) frame. Experimental results are compared with the already existed methods [14,20] and it has shown better values for (PSNR) as associated to other (GIF) Steganography methods. In future, similar extended version of this algorithm could be applied on some different file format like audio and video file format.

Acknowledgement: Dr. Arshiya Sajid Ansari would like to thank Deanship of Scientific Research at Majmaah University for supporting this work under the Project No. R-2021-114.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] T. Morkel, J. H. Eloff and M. S. Olivier, "An overview of image steganography," *Information Security South Africa Conference (ISSA)*, vol. 1, no. 2, pp. 1–11, 2005.

[2] A. Altaay, S. Sahib and M. Zamani, "An introduction to image steganography techniques," in *Proc. Int. Conf. on Advanced Computer Science Applications and Technologies (ACSAT)*, IEEE, Malaysia, pp. 122–126, 2012.

[3] S. Bakhshi, D. Shamma, L. Kennedy, Y. Song and P. D. Juan *et al.*, "Fast, cheap, and good: Why animated GIFs engage us," in *Proc. Chi Conf. on Human Factors in Computing Systems, ACM Digital Library*, San Jose, CA, USA, pp. 575–586, 2016.

[4] N. Tiwari and M. Shandilya, "Evaluation of various LSB based methods of image steganography on GIF file format," *International Journal of Computer Applications*, vol. 6, no. 2, pp. 1–4, 2010.

[5] E. Abed Elgabar and H. Alamin, "Comparison of LSB steganography in GIF and BMP images," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 3, no. 4, pp. 79–83, 2013.

[6] E. Abed Elgabar and F. Mohammed, "JPEG versus GIF images in forms of LSB steganography," *International Journal of Computer Science and Network*, vol. 2, no. 6, pp. 86–93, 2013.

[7] E. Abed Elgabar, "Comparison study of LSB steganography for JPEG and GIF images," *European Academic Research*, vol. 2, no.10, pp. 12776–12785, 2015.

[8] R. Munir, "Application of the modified EzStego algorithm for hiding secret messages in the animated GIF images," in *Proc. 2nd (ICSITech)*, IEEE, Balikpapan, Indonesia, pp. 58–62, 2016.

[9] M. Juzar and R. Munir, "Message hiding in animated GIF using multibit assignment method," in *Proc. (ISESD) IEEE*, Bandung, Indonesia, pp. 225–229, 2016.

[10] K. Miltner and T. Highfield, "Never gonna GIF you up: Analyzing the cultural significance of the animated GIF," *Sage Journals Social Media + Society*, vol. 3, no. 3, pp. 1–11, 2017.

[11] I. Fathurohman, T. Purboyo and R. Nugrahaeni, "Comparative analysis of steganography using LSB and adaptive method on GIF image," *International Journal of Applied Engineering Research*, vol. 12, no. 21, pp. 10999–11006, 2017.

[12] M. Abood, "An efficient image cryptography using hash-LSB steganography with RC4 and pixel shuffling encryption algorithms," in *Proc. Annual Conf. on New Trends in Information & Communications Technology Applications (NTICT) IEEE*, Baghdad, Iraq, pp. 86–90, 2017.

[13] A. Amirulhaqi, T. Purboyo and R. Nugrahaeni, "Security on GIF images using steganography with LSB method, spread spectrum and the vigenere cipher," *International Journal of Applied Engineering Research*, vol. 12, no. 23, pp. 13604–13609, 2017.

[14] R. Basak, K. Dasgupta and P. Dutta, "Steganography in grey scale animated GIF using hash-based pixel value differencing," in *Proc. Fourth Int. Conf. on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Kolkata, India, pp. 248–252, 2018.

[15] M. Hashim, M. Rahim, F. Johi, M. Taha and H. Hamad, "Performance evaluation measurement of image steganography techniques with analysis of LSB based on variation image formats," *International Journal of Engineering & Technology*, vol. 7, no. 4, pp. 3505–3514, 2018.

[16] J. Lin, Z. Qian, Z. Wang, X. Zhang and G. Feng, "A new steganography method for dynamic GIF images based on palette sort," *International Journal of Wireless Communications and Mobile Computing*, Vol. 2020, pp. 1–14, 2020.

[17] Z. Zhu, Q. Ying, Z. Qian and X. Zhang, "Steganography in animated emoji using self-reference," *Multimedia Systems*, vol. 27, no. 3, pp. 331–340, 2021.

[18] A. Gupta, "Steganography with PNG image format: Web supporting media," *International Journal of Research and Analytical Reviews (IJRAR)*, vol. 7, pp. 542–544, 2020.

[19] R. Mstafa, Y. Younis, H. Hussein and M. Atto, "A new video steganography scheme based on Shi-tomasi corner detector," *IEEE Access*, vol. 8, pp. 161825–161837, 2020.

[20] R. Basak, R. Chatterjee, P. Dutta and K. Dasgupta, "Steganography in color animated image sequence for secret data sharing using secure hash algorithm," in *Research Square*, Kolkata, India, pp. 1–18, 2021.

[21] F. Lubis, S. Suwilo and P. Sihombing, "Analysis of LSB algorithm modification with bit inverse and insertion based on length of message," in *Proc. Int. Conf. on Culture Heritage, Education, Sustainable Tourism, and Innovation Technologies (CESIT 2020)*, Setúbal, Portugal, pp. 522–529, 2021.

[22] A. Ansari, M. Mohammadi and M. Parvez, "A Multiple-format steganography algorithm for color images," *IEEE Access8*, Vol. 8, pp. 83926–83939, 2020.

[23] Y. T. Chen, J. J. Tao, L. Y. Liu, J. Xiong and R. L. Xia *et al.*, "Research of improving semantic image segmentation based on a feature fusion model," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, pp. 1–13, 2020.

[24] Y. J. Luo, J. H. Qin, X. Y. Xiang, Y. Tan, Q. Liu *et al.*, "Coverless real-time image information hiding based on image block matching and dense convolutional network," *Journal of Real-Time Image Processing*, vol. 17, no. 1, pp. 125–135, 2020.

[25] A. Ansari, M. Mohammadi and M. Parvez, "A comparative study of recent steganography techniques for multiple image formats," *International Journal of Computer Network and Information Security*, vol. 11, no. 1, pp. 11–25, 2019.

[26] W. You, H. Zhang and X. Zhao, "A siamese CNN for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 291–306, 2020.

[27] H. Ghasemzadeh and M. Kayvanrad, "Comprehensive review of audio steganalysis methods," *Institution of Engineering and Technology (IET Signal Processing)*, vol. 12, no. 6, pp. 673–687, 2018.