Received June 7, 2020, accepted July 1, 2020, date of publication July 14, 2020, date of current version August 5, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3009234*

# Enhanced Least Significant Bit Replacement Algorithm in Spatial Domain of Steganography Using Character Sequence Optimization

**JAGAN RAJ JAYAPANDIYAN[1], C. KAVITHA[2], AND K. SAKTHIVEL[3]**
[1]Department of Computer Science, Periyar University, Salem 636011, India
[2]Department of Computer Science, Thiruvalluvar Government Arts College, Rasipuram 637401, India
[3]Department of Computer Science and Engineering, K. S. Rangasamy College of Technology, Namakkal 637215, India

Corresponding author: Jagan Raj Jayapandiyan (jae.jaganraj@gmail.com)

**ABSTRACT** Text Steganography has become a dominant research field in information sharing domain and many researches are being conducted to strengthen this area. Researches around the amount of secret message that could be stored in a given cover image is always critical for any steganography technique used to share the secret text. This research paper proposes an enhanced Least Significant Bit (eLSB) embedding technique in steganography, through which the quality of cover image is improved, when compared to typical LSB algorithm used in steganography. The proposed method employs in spatial domain and it does the secret message encoding in two phases. The first phase generates the metadata and embeds the header information in first few bytes of cover image and then the following phase takes care of processing secret message and storing the secret message in cover image using an optimized way, which is possible through analyzing secret text's character sequences. Proposed work results into occupying lesser space for the given secret text in cover image and hence leads to the better stego image quality than existing LSB algorithms. As the algorithm works on optimizing secret message during embedding phase itself, this technique enables high capacity embedding rate, additional security due to secret message preprocessing and enhanced cover image quality. The results are compared with LSB algorithm and compared to Peak Signal to Noise Ratio (PSNR), Mean Square Error (MSE) and Root Mean Square Error (RMSE) values to prove the proposed algorithm performs better on secret text embedding in cover image.

**INDEX TERMS** Image steganography, information hiding, secret text sharing, enhanced least significant bit steganography.

## I. INTRODUCTION

Steganography is one of the branches in information hiding [1], which is used for concealed communication. This technique embeds secret message into an any other cover medium such as text, document, image, audio, and video file formats [2]–[7]. Over the last few years, image steganography has drawn extensive interest for its capability of storing huge amount of data. Information security, electronic copyright protection, and other security issues compel people to pay much attention to information hiding technologies. As one of its branches, steganography is to hide the very existence of a secret message in the clandestine communication. As an effectively complementary technology to Cryptography, steganography has drawn tremendous attention of researchers in the recent years.

The science of hiding information in plain sight is called steganography. This technique is also called as "covered writing". Unlike cryptography, steganography's purpose is to obscure the presence of secret message altogether rather than conceal its material [2], [8], [9]. Nowadays, steganography's most common use is to cover one computer file inside another computer file.

The associate editor coordinating the review of this manuscript and approving it for publication was Malik Najmus Saqib.

## II. TYPES OF STEGANOGRAPHY

Steganography technique can be implemented using several cover file formats. The secret information that are being embedded in the cover image also need not be necessarily a text secret message, it can also be a media like contents. Primarily, the steganography types are always defined based on the cover file being used in the embedding phase. The cover file used can be any media format files like image file, audio or video files. Rarely, steganography also takes advantage on the computer protocols and embeds the secret text in it. Steganography techniques can be classified as follows:

### *A. TEXT STEGANOGRAPHY*

In text steganography technique, the cover file used would be in text format and the secret message being embedded in the cover file also would be predominantly of text type. Text steganography embedding strategies are based on number of lines, white spaces, capital letters, like used in radio communication's Morse code.

### *B. AUDIO STEGANOGRAPHY*

Embedding secret messages into digital sound file is known as audio steganography. Steganography methods can embed messages in variety of sound files including.au,.mp3,.wav etc., Human Auditory System (HAS) are exploited in the process of audio steganography. Like the image steganography, this technique also embeds more data into a cover file.

### *C. VIDEO STEGANOGRAPHY*

Video steganography hides the given secret message into a cover object, whose file type is video. In this steganography, video frames are used as carrier source and this cover frame allows the sender to embed a secret message.

### *D. PROTOCOL STEGANOGRAPHY*

Protocol steganography [11] is a new approach for data hiding, which is becoming popular in recent days. In this steganography, network layer protocol of TCP/IP suite are used for data hiding and not restricted only to these protocols. In network layer of OSI model, covert channels are used for data hiding. Covert channels breaches security policies of the network system. These channels are either used to steal the information or communicate secret message over a network using the network protocol. Example protocols used in the protocol steganography are TCP, IPv4, NFS, CIFS etc.,

### *E. IMAGE STEGANOGRAPHY*

Image steganography is a technique in which hiding of a secret message implemented by taking the cover object as the image file. In this steganography, graphic images are commonly used as cover source and this cover image allows the user to embed a huge number of bits from the secret message. The major benefit of image steganography is that an attacker's focus isn't drawn by the cover image.

#### 1) STEGANOGRAPHY PHASES

Any Steganography algorithm must come across following stages to complete its secret message sharing process from a sender to receiver.

Sender: Sender's primary responsibility is to embed the secret message in stego-medium and sending it across communication channel.

Communication Channel: Physical or wireless medium, which carries the encoded cover image with secret message across the network or through any other storage medium. Embedding technique should be mature enough to protect secret message for all possible man in the middle attacks.

Receiver: It is the last phase in this steganography process, where the cover medium is received and then decoded to see what the secret text has been sent over the communication channel

Most of the times, sender augments any strong encryption method and store the encrypted secret message in Cover medium. Below Fig. 1 [10] explains a steganographic process in a typical algorithm.

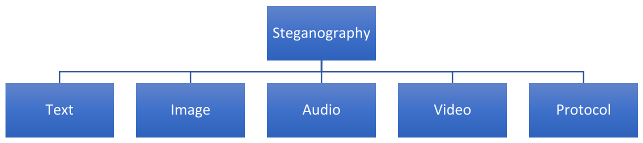


**FIGURE 1. Types of steganography.**

#### 2) LEAST SIGNIFICANT BIT (LSB) REPLACEMENT ALGORITHM

The Least Significant Bit Replacement algorithm is a commonly used straightforward steganographic algorithm, which is used to embed secret message inside a cover medium. In this method, the secret message is stored in the least significant bits of the original cover image and the bits are flipped accordingly. In this process, the change only takes place at the least significant bits [12] of the original image and never on the most significant bits due the resultant noise in the image.

Basic operations involved in any LSB replacement scheme can be mathematically explained as below. Let C be the cover medium being used to embed the secret text. Image C, whose size is X × Y pixels and each pixel has the 24 bit value to store 8 bits on red, green and blue color values as explained in (1).

$$C=\{c_{ij}|0\leq i\leq X, 0\leq i\leq Y, c_{ij}\in\{0,1,2,\ldots,255\}\} \tag{1}$$

To embed the secret message (M) in the given cover image (C), each ASCII character in M has to be converted to its equivalent binary form. For example the character 'A' can be represented as 65 in decimal and (1000 0001)b in binary.

$$M = \{m_i|0\leq i\leq n, s_i\in\{0,1\}\} \tag{2}$$
$$S' = \{s'_i|0\leq i\leq n', s_i\in\{0,1,\ldots,2^{k-1}\}\} \tag{3}$$

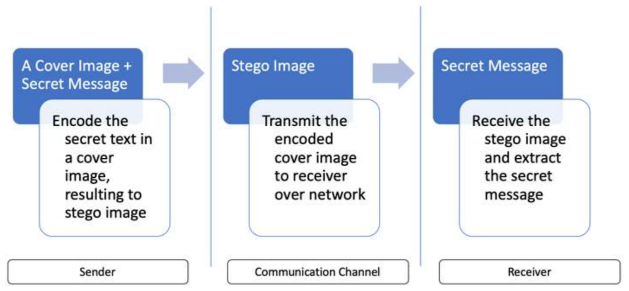

A Cover Image + Secret Message
Encode the secret text in a cover image, resulting to stego image
Stego Image
Transmit the encoded cover image to receiver over network
Secret Message
Receive the stego image and extract the secret message
Sender
Communication Channel
Receiver

**FIGURE 2.** Phases of a typical steganography algorithm.

Secret message (M) can be embedded in each pixel of the cover image (S) by processing the k least significant bits (LSB) in the cover image, C. value k is directly proportional the noise observed after the embedding.

$$S'_i = \sum_{j=0}^{k-1} s_{i\times k+j} \times 2^{k-1-j} \tag{4}$$

Each pixel of the cover image is processed until the secret message is completely embedded into the cover image (4). The quality of the resulting stego image would be degraded if the value of k is increased to improve the embedding capacity of the cover image. This can show the existence of secret text within the cover image. Once the secret messages embedded into the cover image, then the message carrying image is called as stego image.

Y. P. Astuti *et al.*, proposed a steganography algorithm [25], where the secret text is XOR-ed thrice to encrypt the secret text with which no optimization on the amount of data stored are considered. Similarly, O. Elharrouss *et al.*, discussed [26] the possibility of having k least significant to be taken in each color channel for the secret text embedding and more the k value take is inversely proportional to the image quality. B. Vishnu has explored the option of compressing the image [27] and storing them in the edge pixels to get the better visual output after embedding the secret text. This is due to the reason that edge value pixel can accommodate more bits in their respective color channels. M. Y. Elmahi and T. M.Wahbi discussed the option of spreading the secret text bits across cover image [28] so that the randomness can be preserved for additional security, which is related to the work done by Jagan Raj and Prasath in [29]. All these LSB techniques are occupying n number of bytes in the cover image, whose secret text is also n bytes. Optimization over the character sequences during embedding process were never attempted in the previous works. The proposed technique can also be followed by another round of compression algorithm augmentation for next level of security, which stays beyond the scope of this work as the objective here is to enhance the embedding rate by identifying the most repeated character sequences.

## III. PROPOSED ALGORITHM

Proposed algorithm enhances the existing least significant bit algorithm, which is discussed in Section-II.B2. Though there

**TABLE 1.** Character sequence substitution table.

| Character sequence | Decimal in base10 | Binary in base2 | Character sequence | Decimal in base10 | Binary in base2 |
|---|---|---|---|---|---|
| the | 129 | 10000001 | make | 178 | 10110010 |
| be | 130 | 10000010 | can | 179 | 10110011 |
| to | 131 | 10000011 | like | 180 | 10110100 |
| of | 132 | 10000100 | time | 181 | 10110101 |
| and | 133 | 10000101 | no | 182 | 10110110 |
| in | 134 | 10000110 | just | 183 | 10110111 |
| that | 135 | 10000111 | him | 184 | 10111000 |
| have | 136 | 10001000 | know | 185 | 10111001 |
| it | 137 | 10001001 | take | 186 | 10111010 |
| for | 138 | 10001010 | people | 187 | 10111011 |
| not | 139 | 10001011 | into | 188 | 10111100 |
| on | 140 | 10001100 | year | 189 | 10111101 |
| with | 141 | 10001101 | your | 190 | 10111110 |
| he | 142 | 10001110 | good | 191 | 10111111 |
| as | 143 | 10001111 | some | 192 | 11000000 |
| you | 144 | 10010000 | could | 193 | 11000001 |
| do | 145 | 10010001 | them | 194 | 11000010 |
| at | 146 | 10010010 | see | 195 | 11000011 |
| this | 147 | 10010011 | other | 196 | 11000100 |
| but | 148 | 10010100 | than | 197 | 11000101 |
| his | 149 | 10010101 | then | 198 | 11000110 |
| by | 150 | 10010110 | now | 199 | 11000111 |
| from | 151 | 10010111 | look | 200 | 11001000 |
| they | 152 | 10011000 | only | 201 | 11001001 |
| we | 153 | 10011001 | come | 202 | 11001010 |
| say | 154 | 10011010 | its | 203 | 11001011 |
| her | 155 | 10011011 | over | 204 | 11001100 |
| she | 156 | 10011100 | think | 205 | 11001101 |
| or | 157 | 10011101 | also | 206 | 11001110 |
| an | 158 | 10011110 | back | 207 | 11001111 |
| will | 159 | 10011111 | after | 208 | 11010000 |
| my | 160 | 10100000 | use | 209 | 11010001 |
| one | 161 | 10100001 | two | 210 | 11010010 |
| all | 162 | 10100010 | how | 211 | 11010011 |
| would | 163 | 10100011 | our | 212 | 11010100 |
| there | 164 | 10100100 | work | 213 | 11010101 |
| their | 165 | 10100101 | first | 214 | 11010110 |
| what | 166 | 10100110 | well | 215 | 11010111 |

**TABLE 1.** ***(Continued.)*** **Character sequence substitution table.**

| | | | | | |
|---|---|---|---|---|---|
| so | 167 | 10100111 | way | 216 | 11011000 |
| up | 168 | 10101000 | even | 217 | 11011001 |
| out | 169 | 10101001 | new | 218 | 11011010 |
| if | 170 | 10101010 | want | 219 | 11011011 |
| about | 171 | 10101011 | because | 220 | 11011100 |
| who | 172 | 10101100 | any | 221 | 11011101 |
| get | 173 | 10101101 | these | 222 | 11011110 |
| which | 174 | 10101110 | give | 223 | 11011111 |
| go | 175 | 10101111 | day | 224 | 11100000 |
| me | 176 | 10110000 | most | 225 | 11100001 |
| when | 177 | 10110001 | us | 226 | 11100010 |

are few enhanced or redefined LSB algorithms are available, this technique focuses on embedding part of the steganography process and led to reduction of number of bits used in the cover-image for storing secret message. All other spatial domain techniques [25]–[29], use one to one byte representation of secret message, which means each byte in the secret message is embedded as a byte in cover image or the change in position of the embedding based on the selected color model. This proposed algorithm works on the spatial domain of image steganography process primarily focuses on optimizing the way that the secret message messages are being embedded.

## *A. EMBEDDING ALGORITHM*

As part of secret message embedding in given cover image, first the algorithm reads the height (X) and width (Y) of the cover image and collects meta data of a secret message like number of words characters. The primary inputs to the algorithm are cover image (C), secret message (M) and number of least significant bit (k) that must be used during the embedding process.

The proposed algorithm maintains a common set of frequently used character patterns (D). same is used as source for optimization technique during the secret text embedding process as defined in Algorithm-1. This embedding algorithm splits the secret message (M) into a string vector (T).

Table-1 shows the mapping between the character sequences ASCII values and its equivalent binary value.

Each element ($T_{[w]}$) in T is iterated and compared to D. for the matching vector element the secret text to be embedded is substituted with their equivalent binary value as shown in the Table-1. After the character sequence substitution in secret message, the output binary data (S) to be embedded will always be lesser than the original secret text (M) binary data as we have substituted one binary value for two or more-character sequences.

**Algorithm 1** Proposed eLSB Algorithm for Embedding the Secret Message in Cover Image

```
procedure eLSB_embed(C, M, k)
    Read cover image, C
    Read secret message, M
    $X \leftarrow$ Height of the cover image, C
    $Y \leftarrow$ Width of the cover image, C
    $W \leftarrow$ Number of words in the message, M
    $L \leftarrow$ Number of characters in the message, M
    $D \leftarrow$ Hash for the frequent words
    $T \leftarrow$ String vector of words from the secret message, M
    $S \leftarrow$ Secret message vector in binary form, T
    Initialize, $S \leftarrow []$
    for $w \leftarrow 1$ to $W$ in steps of 1 do
        if keyfound($T_{[w]}$)
            $S_{[w]} \leftarrow binary(D\{(T_{[w]})\})$
        else
            $S_{[w]} \leftarrow binary(T_{[w]})$
        end if
    end for
    $H \leftarrow binary(StegoHeader\ (S))$
    $S \leftarrow H + S$
    for $i \leftarrow 1$ to $Y$ in steps of 1 do
     for $j \leftarrow 1$ to $X$ in steps of 1 do
      for $x \leftarrow 1$ to 8 in steps of 1 do
       $rb = resetFromNthBit(k)_b$
       $C = \sum_{l=(8\text{-k})}^{8} (c_{[i][j][x]} \& rb)$
       $C = \sum_{l=(8-k)}^{8} (c_{[i][j][x]} | s_{[c++]})$
      end for
     end for
    end for
  return C, the secret text embedded stego image
end procedure
```

For example, when trying to embed the sample secret message "I have the documents, let me know when can we meet" Any of the typical LSB algorithm will need a 400 bits in cover image to store this text as the number of character bytes in the secret text is 50. Total bits needed in Cover image for embedding is calculated by finding product of the number of ASCII bytes and 8. In any of the LSB technique, we would need 400 bits in the cover image to store this text.

Total needed bits (LSB) $= 50 * 8 => 400$ bits

However, using the proposed eLSB algorithm would only require 35 bytes, which is 280 bits. Refer Table-2 for the byte calculations.

Below Figure-3 shows the sample screenshot from actual implementation of proposed embedding algorithm, which outputs several metrics of the algorithm for a sample secret message and cover image file.

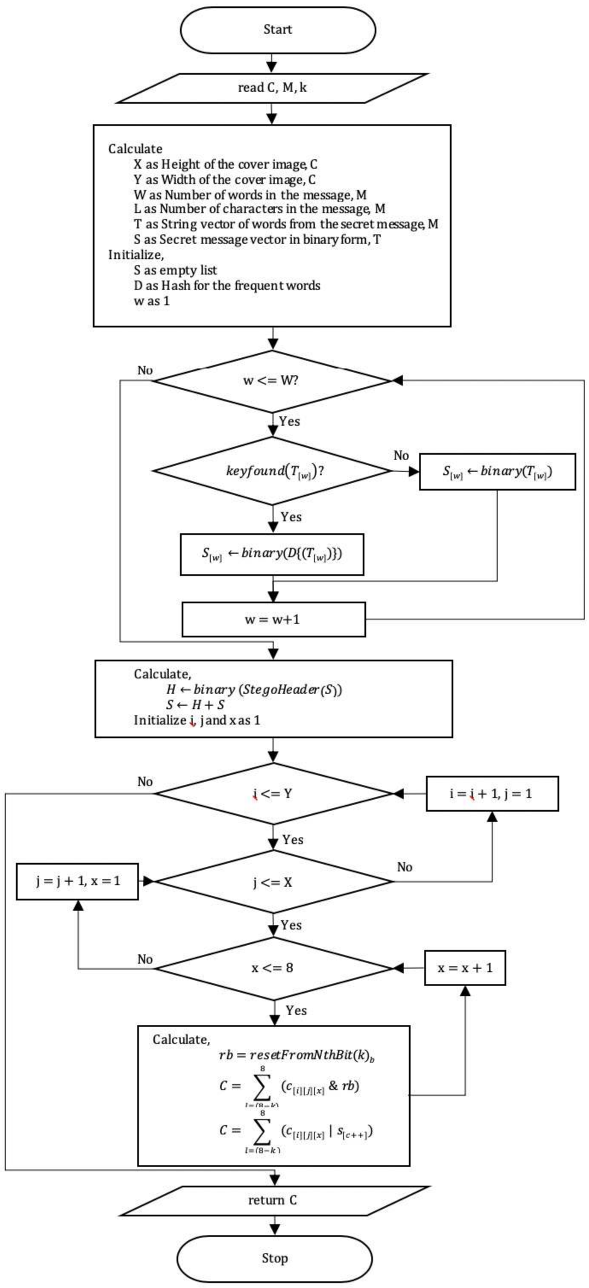

Start
read C, M, k
Calculate
X as Height of the cover image, C
Y as Width of the cover image, C
W as Number of words in the message, M
L as Number of characters in the message, M
T as String vector of words from the secret message, M
S as Secret message vector in binary form, T
Initialize,
S as empty list
D as Hash for the frequent words
w as 1
w <= W?
No
Yes
keyfound(T[w])?
No
S[w] ← binary(T[w])
Yes
S[w] ← binary(D[(T[w])])
w = w+1
Calculate,
H ← binary (StegoHeader(S))
S ← H + S
Initialize i, j and x as 1
i <= Y
i = i + 1, j = 1
j <= X
j = j + 1, x = 1
x <= 8
x = x + 1
Calculate,
rb = resetFromNthBit(k)_b
C = Σ (c[i][j][x] & rb)
C = Σ (c[i][j][x] | s[c++])
return C
Stop

**FIGURE 3.** Flowchart for embedding technique in enhanced least significant bit algorithm.

**TABLE 2.** Sample total byte need calculation and comparison between LSB and eLSB algorithm.

| | Number of Bytes Needed (including trailing spaces) | |
|---|---|---|
| **Word** | **LSB techniques** | **eLSB** |
| I | 2 | 2 |
| have | 5 | 2 |
| the | 4 | 2 |
| documents, | 11 | 11 |
| let | 4 | 4 |
| me | 3 | 2 |
| know | 5 | 2 |
| when | 5 | 2 |
| can | 4 | 2 |
| we | 3 | 2 |
| meet | 4 | 4 |
| **Total** | **50** | **35** |

**TABLE 3.** Numerical illustration of embedding a secret byte in RGB channels of a pixel.

| Color Channel | Color Channel Bytes in a pixel (Before embedding) | | | | | | | | Secret text Byte (S) to insert | Color Channel Bytes in a pixel (after embedding) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Byte-7 | Byte-6 | Byte-5 | Byte-4 | Byte-3 | Byte-2 | Byte-1 | Byte-0 | | Byte-7 | Byte-6 | Byte-5 | Byte-4 | Byte-3 | Byte-2 | Byte-1 | Byte-0 |
| R | 1 | 1 | 0 | 1 | 0 | 1 | 1 | **x** | **1** | 1 | 1 | 0 | 1 | 0 | 1 | 1 | **1** |
| G | 1 | 0 | 0 | 0 | 0 | 0 | 1 | **x** | **0** | 1 | 0 | 0 | 0 | 0 | 0 | 1 | **0** |
| B | 0 | 0 | 0 | 1 | 1 | 1 | 1 | **x** | **1** | 0 | 0 | 0 | 1 | 1 | 1 | 1 | **1** |
| R | 1 | 0 | 1 | 0 | 1 | 1 | 1 | **x** | **1** | 1 | 0 | 1 | 0 | 1 | 1 | 1 | **1** |
| G | 1 | 1 | 0 | 1 | 0 | 0 | 1 | **x** | **0** | 1 | 1 | 0 | 1 | 0 | 0 | 1 | **0** |
| B | 0 | 1 | 0 | 0 | 0 | 1 | 1 | **x** | **1** | 0 | 1 | 0 | 0 | 0 | 1 | 1 | **1** |
| R | 1 | 1 | 0 | 1 | 0 | 1 | 0 | **x** | **0** | 1 | 1 | 0 | 1 | 0 | 1 | 0 | **0** |
| G | 1 | 0 | 1 | 1 | 0 | 1 | 0 | **x** | **1** | 1 | 0 | 1 | 1 | 0 | 1 | 0 | **1** |

Numerical illustration given below explains how each color in a pixel is taken and then the value is embedded. For nullifying the k number of LSB bits in a color byte, each byte is logical AND-ed with (11111110), if the k value is 1. For embedding a bit from secret text byte, the extracted secret bit is again logical OR-ed with color byte's k bits as mentioned in Table-3.

Embedding algorithm also generates a header (H) on the meta data of secret message and embers into first 64 bytes of coverage. This meta data contains information like length of the secret text, k value for the embedding in each LSB etc.,

Once the character substitution has completed, the embedding process stores the optimized secret message's binary data into the least significant bits (LSB) of cover image based on the k value, k defines the number of least significant bits to be used in each byte of the colour values associated to a pixel (5).

$$C = \sum\nolimits_{l=(8-k)}^{8} (c_{[i][j][x]} | s_{[c++]}) \tag{5}$$

## B. EXTRACTION ALGORITHM

Extraction algorithm is similar to compression procedure, but the steps are orderly reversed to obtain original secret message from stego-image. Before processing the secret message, the 64-byte header embedded in the stego-image is read and then meta data is processed. This yields various extraction algorithm variables like height of the image (X), width the image (Y) and the size of the image (l). Each pixel byte ($C_{[w]}$) is iterated and k number of bits are extracted from the stego image, C. Output of the above procedure gives the secret message in binary byte array ($S$).

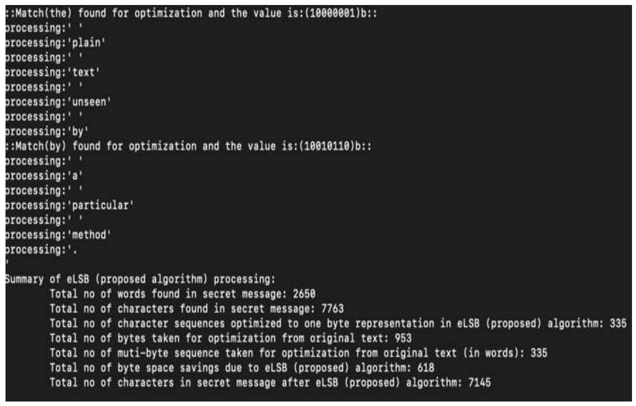
Later, the secret message binary byte array elements $S_{[w]}$ are compared to the frequency dictionary keys, where the keys are binary byte and the values are respective character sequence as given in Table-1 and then translated to respected characters sequence, if it is found else the binary byte will be directly converted to respective ASCII value and stored in a string array $E$.

**Algorithm 2** Proposed eLSB Algorithm for Extracting the Secret Message in Cover Image

```
procedure eLSB_extract(C)
    Read Stego image, C
    $H \leftarrow extracted$ header from stego image, C
    $X \leftarrow Height\ of\ the\ cover\ image\ from\ header, H$
    $Y \leftarrow Width\ of\ the\ cover\ image\ from\ header, H$
    $k \leftarrow number\ of\ LSB\ used\ from\ header\ data, H$
    $l \leftarrow Length\ of\ secret\ message\ in\ bytes\ from$
     $header\ data, H$
    $D \leftarrow$ Hash $for\ the\ frequent\ words$
    $Initialize, h \leftarrow 64$
    $for\ i \leftarrow 1\ to\ Y\ in\ steps\ of\ 1\ do$
      $for\ j \leftarrow 1\ to\ X\ in\ steps\ of\ 1\ do$
        $for\ x \leftarrow 1\ to\ 8\ in\ steps\ of\ 1\ do$
          $if\ bytes \leq h$
            $bytes \leftarrow bytes + 1$
            $continue\ next\ iteration\ in\ i\ loop;$
          $end\ if$
          $T = \sum_{l=(8-k)}^{8} (c_{[i][j][x]} \gg k)$
          $S_{[i]} = S_{[i]} + T$
        $end\ for$
      $end\ for$
    $end\ for$
    $for\ w \leftarrow 1\ to\ l\ in\ steps\ of\ 1\ do$
      $if\ (key\_exists(S_{[w]}))$
        $E_{[w]} \leftarrow D\{(S_{[w]})\}$
      $else$
        $E_{[w]} \leftarrow binary(E_{[w]})$
      $end\ if$
  $end\ for$
  $return\ E, the\ secret\ message$
end procedure
```

## IV. RESULTS AND DISCUSSION

On implementing the above embedding and extraction algorithm results the following tables. After the steganographic



```
::Match(the) found for optimization and the value is:(10000001)b::
processing:' '
processing:'plain'
processing:' '
processing:'text'
processing:' '
processing:'unseen'
processing:' '
processing:'by'
::Match(by) found for optimization and the value is:(10010110)b::
processing:' '
processing:'a'
processing:' '
processing:'particular'
processing:' '
processing:'method'
processing:'.
'
Summary of eLSB (proposed algorithm) processing:
        Total no of words found in secret message: 2650
        Total no of characters found in secret message: 7763
        Total no of character sequences optimized to one byte representation in eLSB (proposed) algorithm: 335
        Total no of bytes taken for optimization from original text: 953
        Total no of muti-byte sequence taken for optimization from original text (in words): 335
        Total no of byte space savings due to eLSB (proposed) algorithm: 618
        Total no of characters in secret message after eLSB (proposed) algorithm: 7145
```

**FIGURE 4.** Sample screenshot of optimized output from embedding algorithm.

**TABLE 4.** Sample images used for the analysis.

| File Name | lena.png_vgilante.txt | monalisa.png_cooldark.txt | dog.png_3wishses.txt | cat.png_fable.txt | lion.png_mike.txt |
|---|---|---|---|---|---|
| LSB | | | | | |
| eLSB | | | | | |

**TABLE 5.** PSNR comparison between LSB and proposed algorithm.

| File Name CoverImage_SecretMessage | PSNR (LSB) | PSNR (eLSB) |
|---|---|---|
| dog.png_3wishses.txt | 65.2515 | 66.1914 |
| cat.png_fable.txt | 61.3756 | 62.2488 |
| lion.png_mike.txt | 77.0321 | 77.2883 |
| lena.png_vgilante.txt | 72.4723 | 74.2795 |
| monalisa.png_cooldark.txt | 50.1151 | 50.4705 |

embedding process, Image quality between original/cover image and stego-image can be measured in several ways. Few important quantitative ways of measuring the image quality between two images are PSNR (Peak Signal Noise Ratio), MSE (Mean Square Error) and RMSE (Root Mean Square Error) [13]. Let's say L is the number of maximum possible intensity levels (minimum intensity level supposed to be 0) in a given image

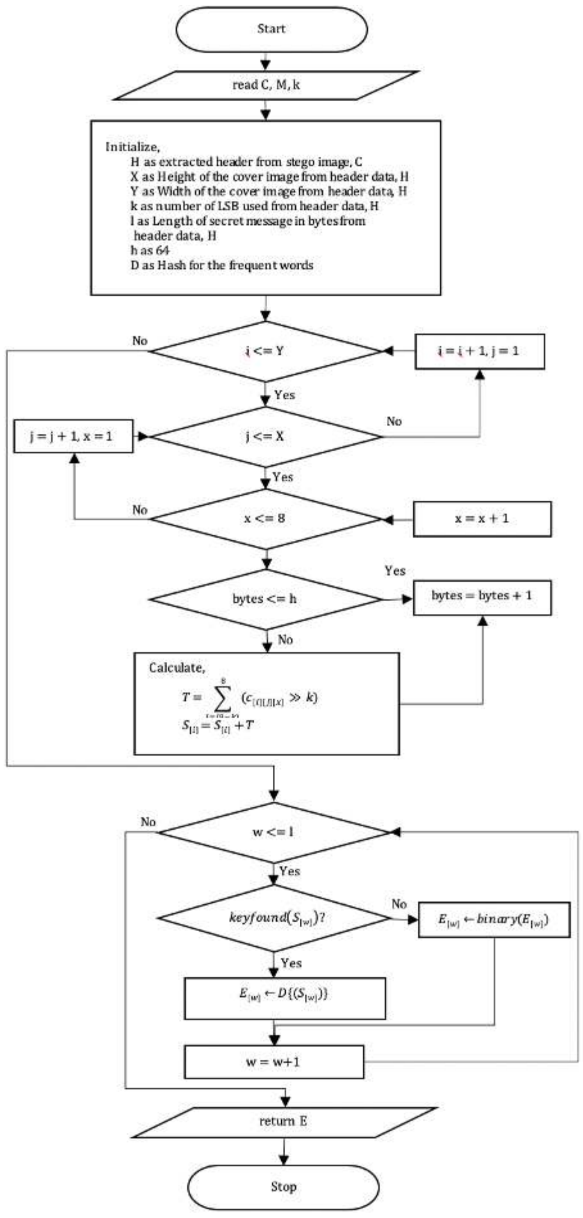

Start
read C, M, k
Initialize,
H as extracted header from stego image, C
X as Height of the cover image from header data, H
Y as Width of the cover image from header data, H
k as number of LSB used from header data, H
l as Length of secret message in bytes from header data, H
h as 64
D as Hash for the frequent words
i <= Y
No
Yes
i = i + 1, j = 1
j <= X
j = j + 1, x = 1
x <= 8
x = x + 1
bytes <= h
bytes = bytes + 1
Calculate,
$T = \sum_{i=(8-k)}^{8} (c_{[i][j][x]} \gg k)$
$S_{[l]} = S_{[l]} + T$
w <= l
keyfound($S_{[w]}$)?
$E_{[w]} \leftarrow binary(E_{[w]})$
$E_{[w]} \leftarrow D\{(S_{[w]})\}$
w = w+1
return E
Stop

**FIGURE 5. Flowchart for extracting technique in enhanced least significant bit algorithm.**

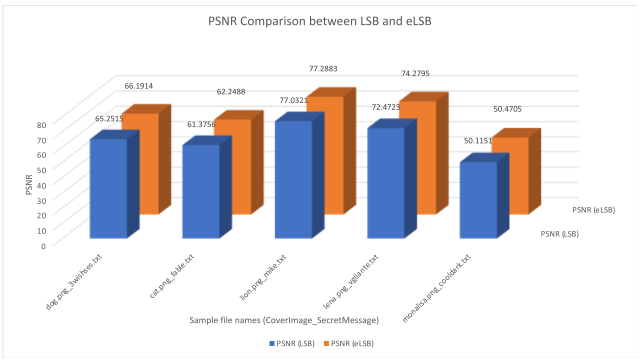


**FIGURE 6. PSNR comparison between LSB and eLSB (proposed algorithm).**

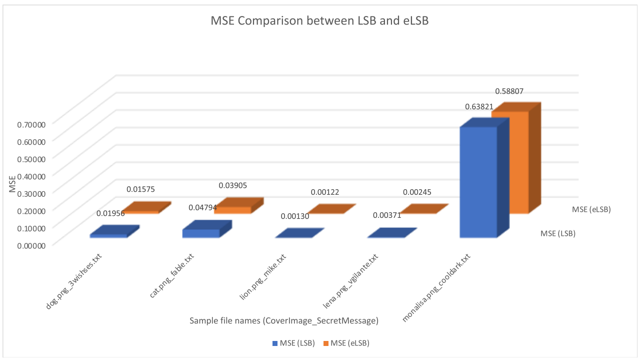


**FIGURE 7. MSE comparison between LSB and eLSB (proposed algorithm).**

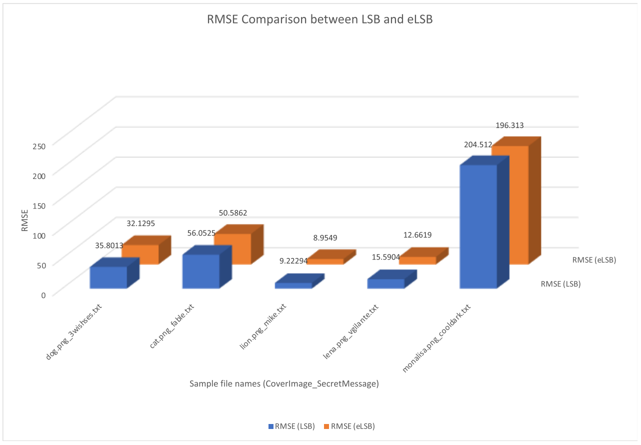


**FIGURE 8. RMSE comparison between LSB and eLSB (proposed algorithm).**

Where, O represents the matrix data of original image. D

$$PSNR = 10\, log_{10}\left(\frac{(L-1)^2}{MSE}\right) = 20\, log_{10}\left(\frac{L-1}{RMSE}\right)$$

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(O(i,j) - D(i,j))^2$$

represents the matrix data of degraded image. m represents the numbers of rows of pixels and i represents the index of that row of the image. n represents the number of columns of pixels and j represents the index of that column of the image. RMSE is the root mean squared error.

On taking sample of five images (Table-4) and five secret messages the below tables explain the PSNR (Table-5), MSE (Table-6), and RMSE (Table-7), of the above proposed algorithm.

Each cover image taken for this experiment went through both LSB and eLSB algorithms for encoding the secret text and measured the image quality after for PSNR, MSE and RMSE.

Comparisons were made between original cover image and LSB stego-image (refer Colum-2 in Table-2) and cover image and eLSB stego-image (refer Colum-3 in Table-3 and Fig.4).

**TABLE 6.** MSE comparison between LSB and proposed algorithm.

| CoverImage_SecretMessage | MSE (LSB) | MSE (eLSB) |
|---|---|---|
| dog.png_3wishses.txt | 0.01956 | 0.01575 |
| cat.png_fable.txt | 0.04794 | 0.03905 |
| lion.png_mike.txt | 0.00130 | 0.00122 |
| lena.png_vgilante.txt | 0.00371 | 0.00245 |
| monalisa.png_cooldark.txt | 0.63821 | 0.58806 |

**TABLE 7.** RMSE comparison between LSB and proposed algorithm.

| CoverImage_SecretMessage | RMSE (LSB) | RMSE (eLSB) |
|---|---|---|
| dog.png_3wishses.txt | 35.8013 | 32.1295 |
| cat.png_fable.txt | 56.0525 | 50.5862 |
| lion.png_mike.txt | 9.22294 | 8.9549 |
| lena.png_vgilante.txt | 15.5904 | 12.6619 |
| monalisa.png_cooldark.txt | 204.512 | 196.313 |

Same exercise is repeated for MSE and RMSE. In PSNR and RMSE having greater value than the LSB algorithm confirms lesser noise in the image. Having lower MSE value tells the image taken is with good quality.

## V. CONCLUSION

Based on the experiment results from Section-IV, it is evident that the proposed eLSB (enhanced Least Significant Bit) replacement algorithm is giving better PSNR, MSE and RMSE values, which in turn conveys the quality of the cover image is undergoing lesser changes compared to any of the traditional LSB algorithm in steganography secret message embedding process. Quantitative analysis between the LSB algorithm and proposed eLSB algorithm are also confirming that the proposed algorithm retains improved image quality across various sizes of image and secret texts. Proposed algorithm also enhances the secret message as they can't be interpreted for the character sequences, which went through the optimization. Future work in the proposed algorithm could be identifying more similar optimizations in the spatial domain and contribute to the better quality of images after embedding.

## REFERENCES

[1] B. Pfitzmann, "Information hiding terminology," in *Proc. ACM 1st Int. Workshop Inf. Hiding*. Berlin, Germany: SpringerVerlag, 1996, pp. 347–350.

[2] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer*, vol. 31, no. 2, pp. 26–34, Feb. 1998.

[3] N. Provos and P. Honeyman, "Hide and seek: An introduction to steganography," *IEEE Secur. Privacy*, vol. 1, no. 3, pp. 32–44, May 2003.

[4] E. Lin and E. Delp, "A review of data hiding in digital images," in *Proc. PICS*, Savannah, GA, USA, 1999, pp. 274–278.

[5] M. M. Sadek, A. S. Khalifa, and M. G. M. Mostafa, "Video steganography: A comprehensive review," *Multimedia Tools Appl.*, vol. 74, no. 17, pp. 7063–7094, Sep. 2015.

[6] F. Djebbar, B. Ayad, K. A. Meraim, and H. Hamam, "Comparative study of digital audio steganography techniques," *EURASIP J. Audio, Speech, Music Process.*, vol. 2012, no. 1, p. 25, Oct. 2012.

[7] H. Singh, P. K. Singh, and K. Saroha, "A survey on text-based steganography," in *Proc. 3rd Nat. Conf.*, New Delhi, India, 2009, pp. 1–6.

[8] W. Mazurczyk, S. Wendzel, S. Zander, A. Houmansadr, and K. Szczypiorski, "Background concepts, definition, and classification," in *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures*. New York, NY, USA: Wiley, 2016.

[9] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[10] J. R. Jayapandiyan, "Optimal secret text compression technique for steganographic encoding by dynamic ranking algorithm," *J. Phys., Conf. Ser.*, vol. 1427, May 2020, Art. no. 012005.

[11] S. Bobade and R. Goudar, "Secure data communication using protocol steganography in IPv6," in *Proc. Int. Conf. Comput. Commun. Control Autom.*, Pune, Maharashtra, 2015, pp. 275–279, doi: 10.1109/ICCUBEA.2015.59.

[12] S. Sugathan, "An improved LSB embedding technique for image steganography," in *Proc. 2nd Int. Conf. Appl. Theor. Comput. Commun. Technol. (iCATccT)*, Bengaluru, Karnataka, 2016, pp. 609–612, doi: 10.1109/ICATCCT.2016.7912072.

[13] Geeks for Geeks. *Python | Peak Signal-to-Noise Ratio (PSNR)*. Accessed: Jan. 2020. [Online]. Available: https://www.geeksforgeeks.org/python-peak-signal-to-noise-ratio-psnr/

[14] R. J. Jagan and S. Prasath, "Validating data integrity in steganographed images using embedded checksum technique," in *Proc. Nat. Conf. Res. Issues Image Anal. Mining Intell. (IJCA NCRIIAMI)*, Jun. 2015, pp. 5–8.

[15] A. Yazdanpanah and M. R. Hashemi, "A new compression ratio prediction algorithm for hardware implementations of LZW data compression," in *Proc. 15th CSI Int. Symp. Comput. Archit. Digit. Syst.*, Tehran, Iran, Sep. 2010, pp. 155–156.

[16] P. A. Alsberg, "Space and time savings through large data base compression and dynamic restructuring," *Proc. IEEE*, vol. 63, no. 8, pp. 1114–1122, Aug. 1975.

[17] Y. Wei, N. Zheng, and M. Xu, "An automatic carving method for RAR file based on content and structure," in *Proc. 2nd Int. Conf. Inf. Technol. Comput. Sci.*, Jul. 2010, pp. 68–72.

[18] S. Khan, M. A. Irfan, M. Ismail, T. Khan, and N. Ahmad, "Dual lossless compression based image steganography for low data rate channels," in *Proc. Int. Conf. Commun. Technol. (ComTech)*, Apr. 2017, pp. 60–64.

[19] I. G. Wiryawan, Sariyasa and I. G. A. Gunadi, "Steganography based on least significant bit method was designed for digital image with lossless compression technique," in *Proc. Int. Conf. Signals Syst. (ICSigSys)*, Bali, Indonesia, 2018, pp. 98–102.

[20] A. Darbani, M. M. AlyanNezhadi, and M. Forghani, "A new steganography method for embedding message in JPEG images," in *Proc. 5th Conf. Knowl. Based Eng. Innov. (KBEI)*, Tehran, Iran, Feb. 2019, pp. 617–621.

[21] K. Rajalakshmi and K. Mahesh, "Video steganography based on embedding the video using PCF technique," in *Proc. Int. Conf. Inf. Commun. Embedded Syst. (ICICES)*, Chennai, India, Feb. 2017, pp. 1–4.

[22] S. L. Chikouche and N. Chikouche, "An improved approach for lsb-based image steganography using AES algorithm," in *Proc. 5th Int. Conf. Electr. Eng. - Boumerdes (ICEE-B)*, Boumerdes, Algeria, Oct. 2017, pp. 1–6.

[23] D. Kaur, H. K. Verma, and R. K. Singh, "A hybrid approach of image steganography," in *Proc. Int. Conf. Comput., Commun. Autom. (ICCCA)*, Noida, Uttar Pradesh, Apr. 2016, pp. 1069–1073.

[24] Y. Yigit and M. Karabatak, "A stenography application for hiding student information into an image," in *Proc. 7th Int. Symp. Digit. Forensics Secur. (ISDFS)*, Barcelos, Portugal, Jun. 2019, pp. 1–4.

[25] Y. P. Astuti, D. R. I. M. Setiadi, E. H. Rachmawanto, and C. A. Sari, "Simple and secure image steganography using LSB and triple XOR operation on MSB," in *Proc. Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Yogyakarta, Indonesia, Mar. 2018, pp. 191–195, doi: 10.1109/ICOIACT.2018.8350661.

[26] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "An image steganography approach based on k-least significant bits (k-LSB)," in *Proc. IEEE Int. Conf. Informat., IoT, Enabling Technol. (ICIoT)*, Doha, Qatar, Feb. 2020, pp. 131–135, doi: 10.1109/ICIoT48696.2020.9089566.

[27] B. Vishnu, L. V. Namboothiri, S. R. Sajeesh, and L. V. Namboothiri, "Enhanced image steganography with PVD and edge detection," in *Proc. 4th Int. Conf. Comput. Methodol. Commun. (ICCMC)*, Erode, India, Mar. 2020, pp. 827–832, doi: 10.1109/ICCMC48092.2020.ICCMC-000153.

[28] M. Y. Elmahi and T. M. Wahbi, "Multi-level steganography aided with compression," in *Proc. Int. Conf. Comput., Control, Electr., Electron. Eng. (ICCCEEE)*, Khartoum, Sudan, Sep. 2019, pp. 1–6, doi: 10.1109/ICCCEEE46830.2019.9071188.

[29] J. Raj and S. Prasath, "Enhancing the data security and data integrity in steganographed images by store bit randomization," *Int. J. Innov. Creative Eng.*, pp. 317–321, Dec. 2015.

**JAGAN RAJ JAYAPANDIYAN** received the master's degree in computer application from Anna University, Chennai, India, in 2009. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Periyar University, Salem, India. He is also a Research Scholar at Periyar University. His major research interests are in information security and steganography domains. He is currently a member of the Association of Computer Machineries (ACM).

**C. KAVITHA** received the Ph.D. degree from Periyar University, Salem, India, in 2008. She is currently working as an Assistant Professor with the Department of Computer Science, Thiruvalluvar Government Arts College, Rasipuram, India. She has published 47 research papers on various national and international journals. Her research interests include image processing, deep learning, and data mining.

**K. SAKTHIVEL** received the Ph.D. degree from Anna University, Chennai, India. He is currently working as a Professor with the Department of Computer Science and Engineering, K. S. Rangasamy College of Technology, Tiruchengode, India. His research interests include image processing and data mining. He is a Lifetime Member of the Indian Society for Technical Education (ISTE).