*2023 4th IEEE Global Conference for Advancement in Technology (GCAT)*
*Bangalore, India. Oct 6-8, 2023*

# A Reversible Data Hiding Technique for Secure Image Transmission

Renji S
*Department of Cyber Security*
*Indian Institute of Information Technology*
Kottayam, India
renjiarun.22phd11006@iiitkottayam.ac.in

Dr. Arun Cyril Jose
*Department of Cyber Security*
*Indian Institute of Information Technology*
Kottayam, India
aruncyril@iiitkottayam.ac.in

2023 4th IEEE Global Conference for Advancement in Technology (GCAT) | 979-8-3503-0525-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/GCAT59970.2023.10353364

***Abstract*—In this paper we propose a methodology for image to text conversion process, which first transforms colour image to grayscale and then performs image to Unicode translation to transform the pixel intensity values into its corresponding Unicode representation for generating a text file. The paper defines an algorithm for the file format conversion process which hides the identity of the image file format as a text file that can later be retrieved back through its inverse transformation procedure. The algorithm also introduces an optimal divisor identification process that improves the efficiency of the methodology. The encoded text file is then Huffman coded to generate a compressed file, which can be decoded to reconstruct the original grayscale image back without much loss of information. The benchmarks such as Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM) attained are as high as 82.25db and 0.9318. Also the Mean Square Error (MSE) has been reduced to as low as 0.0001 respectively, which indicates that the proposed methodology can very well be applied in those application domains where secure transmission of confidential images are highly essential to maintain the integrity of data.**

***Keywords— Decoding, Encoding, Grayscale, Huffman coding, Image reconstruction***

## I. Introduction

Reversible Data Hiding (RDH) [1] is a technique used to embed data into a host medium such as audio, video, image or a text file, from which the embedded data can be retrieved back with unnoticeable variations. It is of great advantage in protecting multimedia contents and is capable of faithfully recovering the original information of high quality. Digital transmissions of images mostly take the form of well-known standard format, the JPEG standard because of its capabilities for sending compressed data without degrading the image quality. However it generates compression artifacts [2], which is unsuitable in those domains that require lossless regeneration of the original information.

This paper proposes a reversible image to text conversion algorithm which converts a grayscale image to its Unicode, represented which can be stored as a text file. The range of the Unicode to which the pixel intensity values are to be translated is predetermined. In this paper the upper and lowercase letters of the English alphabets are chosen as the symbols to which the pixel intensity values are to be encoded. The proposed algorithm also identifies and optimal divisor in the selected range of Unicode based on the maximum gray intensity level of the image. Huffman coding identifies the optimal number of bits used for encoding the alphabets with minimum number of bits as possible for generating a compressed text file. This file can be used for storage as well as for transmission and also to retrieve the original image back with negligible loss of information, thus reconstructing an exact copy of the original image. This overcomes the compression artifacts that occur in pattern embedded images when stored as JPEG images. The pattern embedded images can be converted to text, instead of storing as JPEG file to avoid information loss and for transmission purpose.

The remaining sections are organized as follows: Section II gives a brief overview of the related works in literature, Section III describes the proposed methodology, Section IV introduces the experimental setup for carrying out the proposed work, followed by results and discussions in Section V, and Section VI concludes the paper and explains the future work.

## II. Literature Survey

F. Naz *et al.* [3] introduces an ASCII based dynamic image encryption method with randomized hiding of confidential data in an image to improve the security aspects of embedding data in an image using ASCII codes. But this technique highly depends on the payload capacity of the cover image. S. Weng *et al.* [4] proposed a dynamic RDH for preserving the texture characteristics of an image by dividing the image into different blocks and exploiting the redundancy among the pixel intensity values within the block. Data is embedded into two pixels, the maximum and second largest pixels in the less correlated blocks, while no data is embedded into highly textured blocks. This again depends on the payload capacity of the cover image.

J. Wang *et al.* [5] puts forward an RDH scheme based on histogram shifting technique using multiple histogram modifications for properly utilizing the smooth regions of the cove image for information hiding, thereby increasing the payload (bits per pixel) associated with the cover image. C. Yu *et al.* [6] proposed an RDH in encrypted images by generating hierarchical label maps for the plain text image so as to increase the embedding payload of the cover image. The method still depends on the texture characteristics of the cover image. P. Puteaux *et al.* [7] carried out an extensive survey of for handles encrypted multimedia data using reversible data hiding in encrypted images (RDHEI) and highlights the importance and characteristics of different classes of RDHEI.

Y. Fu *et al.* [8] focuses their attention on RDHEI for privacy-preserving image processing based on adaptive encoding strategy. They accommodate data into the MSB layer of the embeddable blocks which are compressed based on their occurrence frequency with reversed Huffman code words, a variation of which was proposed by Z. Yin *et al.* [9], which introduces an RDHEI algorithm of high capacity by predicting the MSB of each pixel for embedding additional information by multi-MSB substitution.

S. Gull *et al.* [10] proposed an RDH technique which preprocesses the secret data using Huffman encoding strategy with dual images. The Huffman code book generated is divided into two parts, which are then embedded into two images that are visually similar to generate dual stego images. C. C. Chen *et al.* in their paper [11] utilizes the higher order nibbles for performing the RDHEI based on Huffman coding by taking the difference between the nibbles of adjacent pixels, thus exploiting the special correlation that exists among the pixel intensity values. The difference of the nibbles among the adjacent pixels is negligible for highly correlated images and is mapped to a narrow range of values and hence the authors compress it efficiently using Huffman coding, creating a large space for data hiding. However, this method highly depends on the special correlation among the adjacent pixel values.

## III. METHODOLOGY

The proposed methodology proposes an encoder and decoder section for reversible image to text conversion process. The image to text encoder is used to encode a grayscale image encrypted as ASCII codes on to a text file, which is further compressed using Huffman coding, from which the original grayscale image can be retrieved back through the inverse process without much loss of information. This is elaborated in Fig. 1.

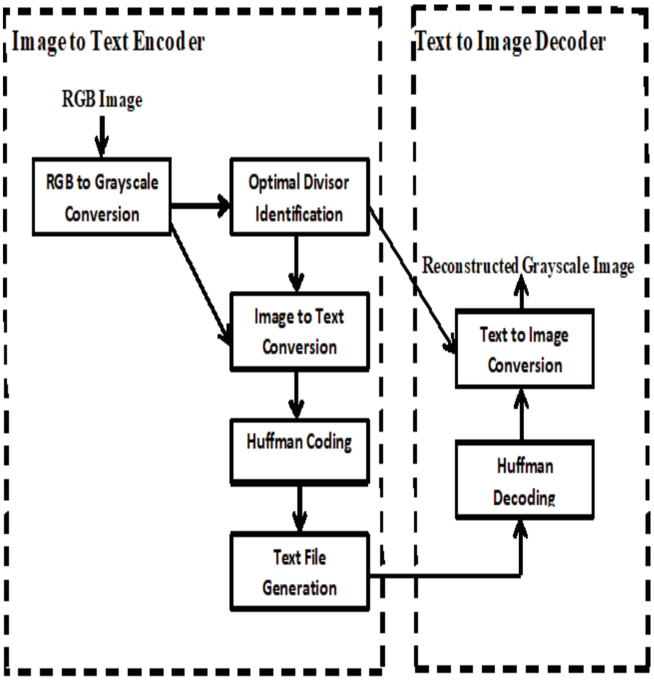



Fig. 1. Proposed Methodology

### A. Image to Text Encoder

The image to text encoder takes as input an RGB image which is converted to its corresponding grayscale image for further processing. The grayscale image is then converted to text file by encoding the pixel intensity values to Unicode characters as mentioned in Algorithm 1.

The methodology focuses on converting grayscale intensity values to upper and lower case ASCII range of English alphabets that spans 65 to 91 for ‘A’ to ‘Z’ and 97 to 122 for ‘a’ to ‘z’, which is the same as that of Unicode representation, except that Unicode is a sixteen bit code compared to ASCII which is an eight bit code. Unicode has been adopted in this paper for making it scalable for other languages as well. Images are first converted into its corresponding ASCII code, pixel by pixel as mentioned in Algorithm 1. This generates a text file which can be used for storage as well as for transmission. The image can be reconstructed from the text file by applying the reverse process.

---

**Algorithm 1** *: Image_to_Text_Encoder(RGB Image)*

---

*Input: RGB Image of size MxN*

*Output: Text file*

1. *Img = read(RGB image)*
2. *Convert Img to grayscale image, G.*
3. *Optimal Divisor (OD) identification: Find optimal divisor for image to text conversion process as follows:*
   i. *Let L be the number of gray levels, say 256.*
   ii. *N= L/n, where n=52/2, half of the total number of upper and lower case letters in the English alphabet.*
   iii. *OD=floor(N)*
4. *Open ‘F.txt’ in write mode.*
5. *Initialize i=1, j=1*
6. *Until i<= M*
7. *Until j<= N*
   i. *Quotient, Q=G(i,j)/OD*
   ii. *Remainder, R = G(i,j) % OD*
   iii. *if Q <= 26,*
      *Q = Q+65*
      *else, Q = Q+70*
   iv. *R = R + 65*
   v. *Q1 = Unicode (Q) and R1=Unicode (R)*
   vi. *Goto step 7*
8. *Goto step 6.*
9. *Write the Q1 and R1 into file ‘F’*
10. *Close the file ‘F’*

---

Algorithm 1 converts the grayscale image by transforming its intensity values to ASCII code by utilizing an optimal divisor identification technique that identifies an optimal divisor for mapping the image intensity values from 0-255 into the ASCII range 65-91 and 97-122. The ASCII codes are then written into a text file, which contains the image data as upper and lower case English alphabets. Huffman coding is then applied to the generated file to ensure efficient compression by assigning fewer bits for most frequently occurring characters and more bits for infrequent characters in the text file.

### *B. Optimal Divisor Identification*

The optimal divisor identification technique identifies the most promising divisor for the specified ASCII range by dividing the total grayscale intensity levels, 'L' by half of the total ASCII range to which the image intensity values are to be mapped. The OD gives the best values of benchmarks, such as PSNR, MSE and SSIM for the proposed methodology. The OD evaluated for this analysis with a total ASCII range of 52, and 26 upper and lower case letters is 9, which gives an average SSIM, PSNR and MSE of 0.9, 76.07db and 0.003 respectively.

### *C. Text to Image Decoder*

To restore the image from text file, perform the steps mentioned in Algorithm 2.

---

**Algorithm 2** : *Text_to_Image_Decoder(Text File)*

---

*Input: Text File*

*Output: Grayscale Image*

1. *Open 'F.txt' generated in Algorithm 1 in read mode.*
2. *Initialize i=0, j =1*
3. *Repeat until end of file (EOF) is reached:*
   - *i.* *i = i+1*
   - *ii.* *Read, C = char(i)*
   - *iii.* *i = i+1*
   - *iv.* *Read R = char(i)*
   - *v.* *R1= ASCII(R), R11=R1 – 65*
   - *vi.* *Q1= ASCII(C)*
   - *vii.* *If Q1>=97*

     *Q1 = Q1 - 70*

     *else*

     *Q1 = Q1 - 65*
   - *viii. Pixel, P(j) = (Q1*OD) + R1*
   - *ix. Grayscale image, Img(j)=P(j)*
   - *x.* *j = j + 1, goto step 3.*

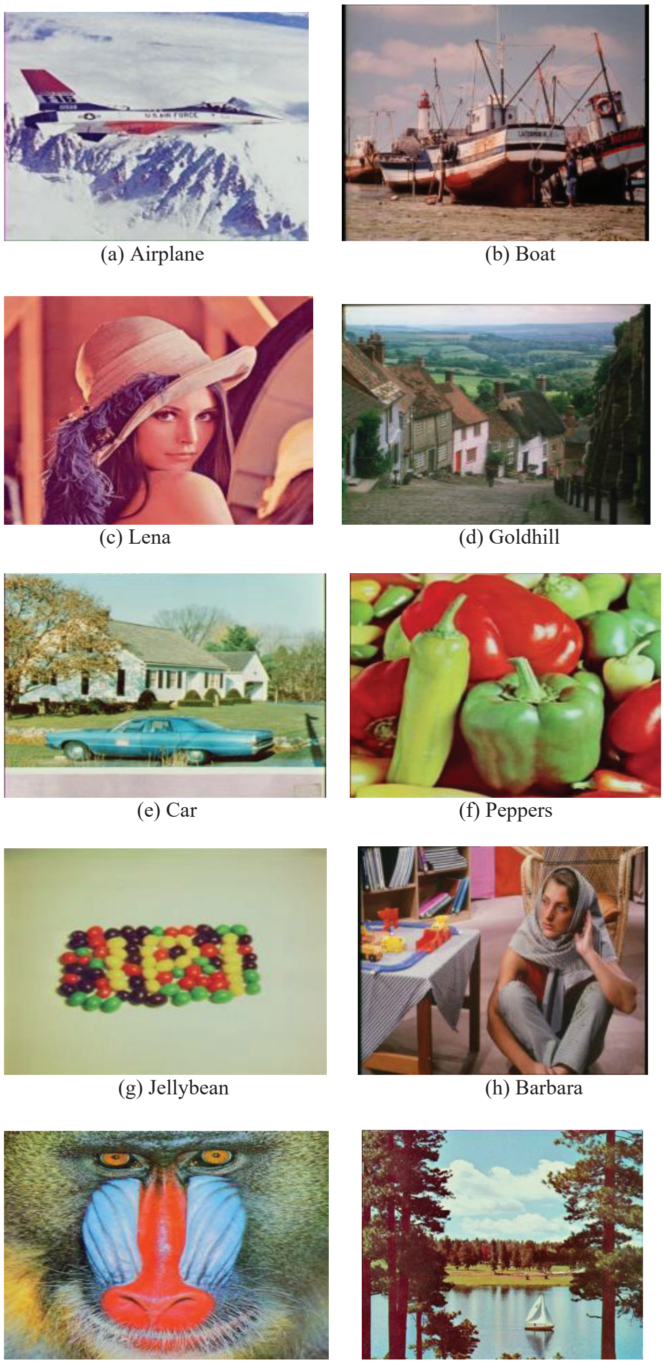
*4. Display the grayscale image, Img.*

---

Algorithm 2 takes a input the encoded text file that stores the upper and lower case English alphabets, which in this case are the letters from 'A' to 'I', since the optimal divisor generated by the algorithm is nine. The quotient as well as the remainder stored into the text file as alphabets are retrieved and then converted to its corresponding pixel intensity value by the reverse process. This generates the grayscale image back from the text file.

## IV. EXPERIMENTAL SETUP

Images were chosen from the miscellaneous volume set of the University of South California-Signal and Image Processing Institute (USC-SIPI) Image Database [13], which includes colour images of size 256 x 256, 512 x 512, and 1024 x 1024. Some of the sample images chosen from SIPI image dataset are shown in Fig. 2. In this paper images have been resized to 320x240 for experimental purpose.



(a) Airplane (b) Boat (c) Lena (d) Goldhill (e) Car (f) Peppers (g) Jellybean (h) Barbara (i) Mandrill (j) Sailboat

Fig. 2. Original Images

## V. RESULTS AND DISCUSSIONS

### *A. Performance Evaluation Metrics*

The proposed method can be evaluated with the help of image quality assessment indicators as mentioned below:

*1) Mean Square Error (MSE)*

MSE indicates the difference between the corresponding intensity values of the original image f(i, j) of size M x N and compressed intensity values of each pixel in the reconstructed image, F(i, j), expressed as[15]:

$$MSE = \frac{1}{M\,x\,N}\sum_{i=1}^{M}\sum_{j=1}^{N}\big(f(i,j) - F(i,j)\big)^2 \quad (1)$$

*2) Peak Signal-to-Noise Ratio (PSNR)*

The quality of compressed images can be estimated using the standard metric, PSNR, which for an 8 bits per picture element of the key image can be expressed as[15]:

$$\text{PSNR (dB)} = 10log_{10}\left(\frac{255^2}{MSE}\right) \quad (2)$$

*3) SSIM*

SSIM [15] is based on three factors, the luminance, contrast and structure of the image to better suit the human visual system. The Matlab SSIM function is made use of in computing the structural similarity between the original and the reconstructed images.

$$\text{SSIM(x,y)} = \frac{(2\mu_x\mu_y+\text{c1})(2\sigma_{xy}+\text{c2})}{(\mu_x^2\mu_y^2+\text{c1})(\sigma_x^2\sigma_y^2+\text{c2})} \quad (3)$$

Here, $\mu_x$ and $\mu_y$ are the mean intensity values and $\sigma_x$ and $\sigma_y$ denotes the standard deviation for computing the luminance, contrast and structure comparison function, for computing the SSIM. The constants, c1=c2 = 0, which corresponds to the "universal quality index" (UQI) [16].

## *B. Generated Text File*

The text file generated for the Lena image based on Algorithm 1 is shown in Fig. 3. The contents of the text file are the pixel intensity values of the grayscale image which has been converted to the upper and lower case English alphabets. The optimal divisor guarantees that the repetition of characters is high, so that Huffman coding generates a compressed file for efficient storage and transmission.

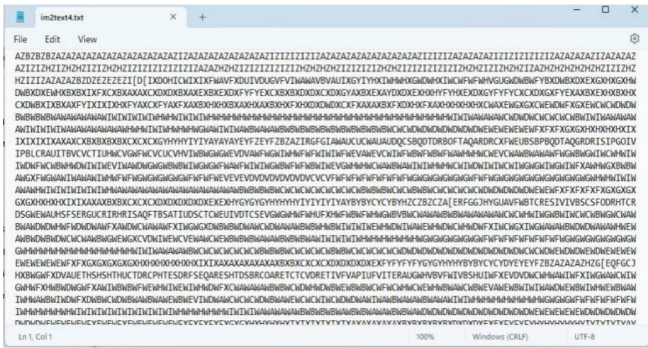



Fig. 3. Generated Text File for Lena Image

The optimal divisor value of nine generates only the upper case letters in the English alphabet, thus increasing the probability of the frequency of occurrence of fewer characters for efficient generation of the Huffman code book as shown in Table I.

## *C. Reconstructed Images from Text File*

The reconstructed grayscale images from the text file are shown in Fig. 4. Comparing the original and reconstructed grayscale images, it is clear that the subjective fidelity criteria have been met, since the two images are visually identical. Further, results in Table I shows that the objective fidelity criteria have also been met, which is evident from the benchmarks attained.

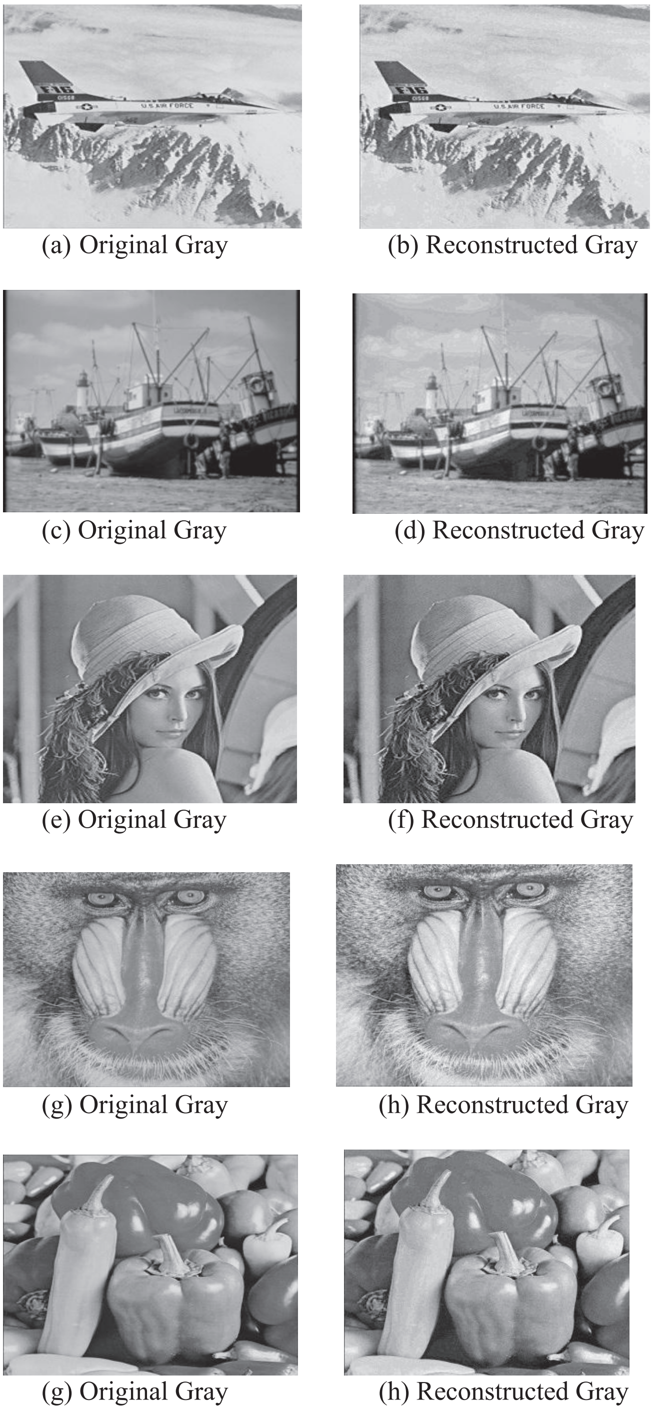



Fig. 4. Original Vs. Reconstructed Grayscale Images

## *D. Table of results*

Table I shows the Huffman code book or the Huffman dictionary generated for the Lena image in the SIPI image dataset. Bits have been allotted based on the probability of occurrence of each alphabet in the text file generated in Algorithm 1.

TABLE I. HUFFMAN CODE BOOK

| Alphabet Code | ASCII Value | Huffman coded bits | Alphabet Code | ASCII Value | Huffman coded bits |
|---|---|---|---|---|---|
| A | 65 | 1 0 1 1 | N | 79 | 0 1 0 0 0 |
| B | 66 | 1 0 1 0 | O | 80 | 0 0 1 0 1 |
| C | 67 | 1 0 0 1 | P | 81 | 0 0 0 0 1 |
| D | 68 | 0 1 1 1 | Q | 82 | 0 0 0 0 0 |
| E | 69 | 0 0 0 1 | R | 83 | 1 1 0 1 0 |
| F | 70 | 1 1 1 | S | 84 | 0 0 1 0 0 0 |
| G | 71 | 0 0 1 1 | T | 85 | 0 0 1 0 0 1 0 |
| H | 72 | 0 1 1 0 | U | 86 | 1 0 0 0 0 1 |
| I | 73 | 0 1 0 1 | V | 87 | 0 1 0 0 1 1 |
| J | 74 | 0 1 0 0 1 0 | W | 88 | 1 0 0 0 0 0 |
| K | 75 | 1 1 0 0 1 | X | 89 | 0 0 1 0 0 1 1 0 |
| L | 76 | 1 0 0 0 1 | Y | 90 | 0 0 1 0 0 1 1 1 0 |
| M | 77 | 1 1 0 1 1 | Z | 91 | 0 0 1 0 0 1 1 1 1 0 |

Table II shows the benchmarks attained for images in the SIPI image dataset, with an optimal divisor value of 9, evaluated in step 3 of Algorithm 1.

TABLE II. BENCHMARKS ATTAINED( OD = 9)

| *Image* | *PSNR* | *MSE* | *SSIM* |
|---|---|---|---|
| Car | 80.35 | 0.0006 | 0.9318 |
| Sailboat | 80.55 | 0.00057 | 0.9187 |
| Airplane | 82.25 | 0.00039 | 0.9001 |
| Boat | 71.51 | 0.0046 | 0.8909 |
| Lena | 78.36 | 0.00096 | 0.8842 |
| Peppers | 75.84 | 0.0017 | 0.8872 |
| Mandrill | 69.46 | 0.0074 | 0.9195 |
| Barbara | 75.46 | 0.0019 | 0.9245 |
| Goldhill | 78.06 | 0.001 | 0.9097 |
| House | 79.16 | 0.00079 | 0.8585 |
| Jellybean | 65.85 | 0.017 | 0.8606 |

The car image in Table II attains a PSNR of 80.35db, with a MSE and SSIM of 0.0006 and 0.9318, in contrast to the jellybean image that shows the benchmark parameters to be 65.85db, 0.017 and 0.8606, which is still an acceptable value for generating good quality images.

The optimal divisor generates the peak value for the benchmarks which was experimented during the analysis with various values in the chosen ASCII code range. Values that were greater or lesser than the optimal divisor value showed decreasing values of the benchmarks, such as PSNR and SSIM and high MSE, which indicates that the procedure applied in the proposed methodology for generating the optimal divisor gives the best results for the reversible image to text conversion process.

TABLE III. BENCHMARKS ATTAINED (OD = 10)

| *Image* | *PSNR* | *MSE* | *SSIM* |
|---|---|---|---|
| Car | 78.08 | 0.001 | 0.914 |
| Sailboat | 80.35 | 0.0006 | 0.9049 |
| Airplane | 80.88 | 0.0005 | 0.8781 |
| Boat | 69.84 | 0.0068 | 0.8646 |
| Lena | 79.3 | 0.0007 | 0.8689 |
| Peppers | 73.78 | 0.0027 | 0.8633 |
| Mandrill | 68.33 | 0.0096 | 0.9007 |
| Barbara | 73.58 | 0.0029 | 0.9079 |
| Goldhill | 79.31 | 0.00076 | 0.8954 |
| House | 77.91 | 0.0011 | 0.8525 |
| Jellybean | 66.25 | 0.0155 | 0.8673 |

Table III shows the benchmarks attained with OD=10, which clearly shows that the benchmarks attained are not as good as those values obtained with an OD value of 9.

TABLE IV. BENCHMARKS ATTAINED ( OD =5)

| *Image* | *PSNR* | *MSE* | *SSIM* |
|---|---|---|---|
| Car | 55.82 | 0.17 | 0.3493 |
| sailboat | 53.01 | 0.33 | 0.2851 |
| Airplane | 57.2 | 0.12 | 0.4719 |
| Boat | 53.23 | 0.31 | 0.4008 |
| Lena | 53.1 | 0.32 | 0.3451 |
| Peppers | 53.04 | 0.32 | 0.3468 |
| Mandrill | 54.07 | 0.26 | 0.2042 |
| Barbara | 52.59 | 0.36 | 0.2224 |
| Goldhill | 51.97 | 0.42 | 0.2377 |
| House | 54.31 | 0.24 | 0.5273 |
| Jellybean | 56.5 | 0.15 | 0.6688 |

Table IV shows the benchmarks attained with OD=5. The benchmarks attained clearly indicate that the choice of OD=5 results in poor reconstruction of the original image from the text file because of the increase in MSE and decrease in SSIM and PSNR values obtained. Thus from the experimental analysis the optimal value for OD evaluated based on the proposed methodology is nine, which shows better values for the benchmarking parameters.

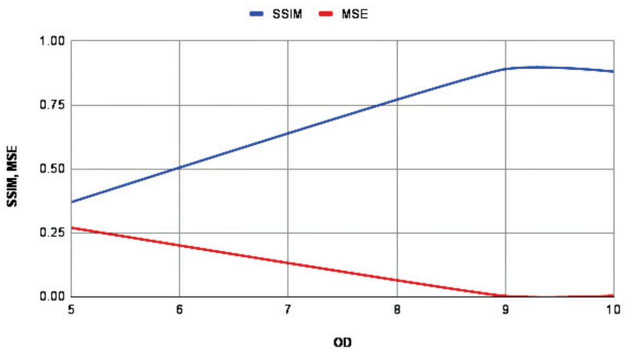



Fig. 5. SSIM and MSE for different values of OD

Fig. 5 shows the variation in SSIM and MSE when OD takes the value of 5, 9, and 10. From the graph it is evident that the mean MSE is high for OD=5 and decreases to zero when OD=9 and then starts increasing again at OD=10. Fig. 6 shows the variation in the mean value of PSNR for the same values of OD.

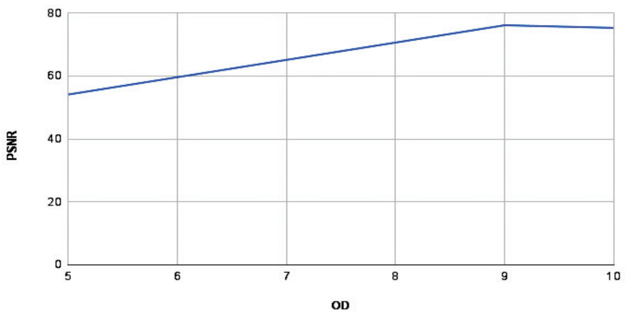



Fig. 6. PSNR for different values of OD

From Fig 6 it is clear that the mean PSNR value reaches its maximum at OD=9 and then starts decreasing again.

## VI. CONCLUSION

The proposed methodology performs an efficient and reversible image to text conversion process using minimum number of ASCII symbols generated using the optimal divisor generation techniques for efficient encoding of the text file using Huffman coding. From the benchmarks attained, it can concluded that the methodology can be extended for visualizing image data in the form of text and vice-versa. At present, the paper focuses on Unicode symbols or ASCII codes for upper and lower case letters in the English alphabet, but it can be extended further to include more number of symbols, including alphabets of different languages encoded by Unicode. Hence the paper provides future directions to further translate an image to symbols in various other languages, paving the way for image to multi-language translation process.

## ACKNOWLEDGMENT

We thank the organization for providing the necessary facilities for carrying out this research work. This work was supported by Phase - II of the Visvesvaraya PhD Scheme for Electronics & IT, Ministry of Electronics & Information Technology (MeitY), Government of India.

## REFERENCES

[1] Y. Jia, Z. Yin, X. Zhang, and Y. Luo, "Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting," *Signal Processing*, vol. 163, pp. 238–246, 2019, doi: 10.1016/j.sigpro.2019.05.020.

[2] J. Liu, S. Member, D. Liu, S. Member, and W. Yang, "A Comprehensive Benchmark for Single Image Compression Artifact Reduction," *IEEE Transactions on Image Processing*, vol. 29, pp. 7845–7860, 2020.

[3] F. Naz, I. A. Shoukat, R. Ashraf, U. Iqbal, and A. Rauf, "An ASCII based effective and multi-operation image encryption method," *Multimed. Tools Appl.*, vol. 79, no. 31–32, pp. 22107–22129, 2020, doi: 10.1007/s11042-020-08897-4.

[4] S. Weng, Y. Q. Shi, W. Hong, and Y. Yao, "Dynamic improved pixel value ordering reversible data hiding," *Inf. Sci. (Ny).*, vol. 489, pp. 136–154, 2019, doi: 10.1016/j.ins.2019.03.032.

[5] J. Wang, X. Chen, J. Ni, N. Mao, and Y. Shi, "Multiple Histograms-Based Reversible Data Hiding: Framework and Realization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2313–2328, 2020, doi: 10.1109/TCSVT.2019.2915584.

[6] C. Yu, X. Zhang, X. Zhang, G. Li, and Z. Tang, "Reversible Data Hiding with Hierarchical Embedding for Encrypted Images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 451–466, 2022, doi: 10.1109/TCSVT.2021.3062947.

[7] P. Puteaux, S. Y. Ong, K. S. Wong, and W. Puech, "A survey of reversible data hiding in encrypted images – The first 12 years," *J. Vis. Commun. Image Represent.* vol. 77, no. January, 2021, Elsevier doi: 10.1016/j.jvcir.2021.103085.

[8] Y. Fu, P. Kong, H. Yao, Z. Tang, and C. Qin, "Effective reversible data hiding in encrypted image with adaptive encoding strategy," *Inf. Sci. (Ny).*, vol. 494, pp. 21–36, 2019, doi: 10.1016/j.ins.2019.04.043.

[9] Z. Yin, Y. Xiang, and X. Zhang, "Reversible Data Hiding in Encrypted Images Based on Multi-MSB Prediction and Huffman Coding," *IEEE Trans. Multimed.*, vol. 22, no. 4, pp. 874–884, 2020, doi: 10.1109/TMM.2019.2936314.

[10] S. Gull, S. A. Parah, and K. Muhammad, "Reversible data hiding exploiting Huffman encoding with dual images for IoMT based healthcare," *Comput. Commun.*, vol. 163, no. August, pp. 134–149, 2020, doi: 10.1016/j.comcom.2020.08.023.

[11] C. C. Chen, C. C. Chang, and K. Chen, "High-capacity reversible data hiding in encrypted image based on Huffman coding and differences of high nibbles of pixels," *J. Vis. Commun. Image Represent.*, vol. 76, no. February, p. 103060, 2021, doi: 10.1016/j.jvcir.2021.103060.

[12] S. Wan, Y. Xia, L. Qi, Y. Yang, and M. Atiquzzaman, "Automated Colorization of a Grayscale Image with Seed Point Propagation," *IEEE Transactions on Multimedia*, vol. 9210, no. c, pp. 1–12, 2020, doi: 10.1109/TMM.2020.2976573.

[13] https://sipi.usc.edu/database/database.php?volume=misc

[14] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Process. Lett.*, vol. 9, no. 3, pp. 81–84, March 2002, doi: 10.1109/97.995823.

[15] D.R.I. M. Setiadi, "PSNR vs SSIM: imperceptibility quality assessment for image steganography," *Multimed. Tools Appl.*, vol. 80, no. 6, pp. 8423–8444, Springer 2021, doi: 10.1007/s11042-020-10035-z.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh,E.P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions On Image Processing*, Vol. 13, No. 4, April 2004