International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-5, March 2020

Improved Data Hiding Scheme using VQ based Index Map Compression and Codebook Extension

P. Uma, S. Vimala

Abstract: *Now-a-days, data protection has become inevitable for confidential transactions that happen over net. Transmission of such confidential data has become a challenging issue in today's scenario. Many Data Hiding techniques are used for transmitting data in a secured way. The secret data can be hidden as part of any type of file such as Text file, Sound file, image file, Video file, etc. It has been proposed to hide secret data as part of cover image. The proposed work adopts Vector Quantization (VQ) which is one of the powerful and simple image compression techniques to compress the size of the cover image and to reduce the cost associated with storage/transmission. VQ transforms the cover image into its corresponding Codebook and IndexMap. The confidential data is then embedded as part of the Codebook and IndexMap. The proposed method helps in improving the performance by increasing the embedding capacity and coding efficiency. The performance of Steganography is improved in three levels. The embedding capacity of the proposed method is increased by 23,726 bits when compared to that of existing similar methods, which is a significant improvement.*

Keywords: Data hiding, Embedding Capacity, Steganography, Vector Quantization.

I. INTRODUCTION

Steganography is a technique that enables us to transmit secret data in secured way by embedding the data as part of any media file such as Audio, Video or Image file. In the proposed method, it has been decided to embed the data as part of the image file. The image used for embedding the secret data is called a Cover image. Relatively large amount of data can be hidden using such techniques [1]. Generally image occupies more storage space and needs more time for transmission. To reduce the cost associated with storage and transmission of Stego-image, many image compression techniques such as VQ [2], Side Match Vector Quantization (SMVQ) [3], JPEG [4], and JPEG2000 [5] are used. Due to its simplicity and cost effective implementation, VQ finds its application in Steganography [6]. VQ based techniques such as Side Match Vector Quantization (SMVQ) are also used for data hiding [7,8,9,10,11,12,13,14,15]. In Vector Quantization, the input image is transformed into Codebook and IndexMap. The secret data can be hidden as part of Codebook and IndexMap without causing much distortion to the image.

VQ comprises of three stages namely 1. Codebook Generation, 2.Image encoding and 3.Image decoding. The quality of the codebook that is generated improves the performance of VQ. In codebook generation phase, the input image is divided into blocks of size 4x4 pixels, which is an optimal size to maintain trade-off between quality and bit-rate of compressed images. Each block is transformed into a vector of size 16 elements $(\{c_1, c_2, \ldots, c_{16}\})$. Collection of such vectors is called Training Set (TS of size N). The desired set (of size M) of training vectors, called the representative vectors is selected from the Training Set and the collection of such vectors is called the Codebook. From the total set of vectors, representative vectors called codevectors are selected to form the codebook of size M, where $M<N$. In image encoding phase, each input block is replaced with index i of the closest codevector c_i. These indices collectively form the IndexMap. The distance between the codevector and the input vector is computed using the Euclidean distance measure using (1).

$$d(x, y_i) = \sum_{j=1}^{k} (x_j - y_{ij})^2 \qquad (1)$$

In equation (1), x_j is the j^{th} component of the input block x (training vector) and y_{ij} is the j^{th} component of the i^{th} codevector y_i. The codevector with which the minimum distance obtained is identified as the similar codevector.

$$d(x, y_i) = \min\{d(x, y_i), y_i \in C\} \qquad (2)$$

where y_i is the most similar codevector to the input vector x. During the decoding phase, each index in the IndexMap is replaced with the corresponding codevector [16].As a result of VQ, the cover image is converted into a set of $CI = \{CB, IM\}$ where CI is the Compressed Image, a set of Codebook (CB) and an IndexMap (IM).

In the proposed method, the CB and IM are used for embedding the secret data as part of them. In general data hiding technique, the secret data is hidden as part of cover image, where the cover image takes more storage space that leads to extra overhead. In addition to embedding the secret data as part of cover image, VQ compresses the cover image that leads to reduction in storage cost. In section 2, similar works are discussed. In Section 3, proposed work is explained. In Section 4, Experimental results are discussed and in Section 5, the conclusion follows.

Revised Manuscript Received on March 30, 2020.
* Correspondence Author
P. Uma*, Research Scholar, Department of Computer Science, Mother Teresa Women's University, Kodaikanal, India. Email: umapoomi@yahoo.com
S. Vimala, Associate Professor, Department of Computer Science, Mother Teresa Women's University, Kodaikanal, India. Email: vimalaharini@gmail.com

II. RELATED WORKS

There are many techniques available to generate codebook [17, 18, 19, 20, 21]. One such technique is Simple Codebook Generation (SCG) [22]. The codebook generated using SCG is improved in terms of quality, using K-Means algorithm. The IndexMap created as a result of Image Encoding is further improved by incorporating the Search Order Coding (SOC) method to improve the coding efficiency [23].

In 2004, the steganography method proposed by Chang et al. [24] is the first one in which the VQ based SOC is used. This method faces the challenge of increased file size and limited capacity. In 2007, Chang et al. proposed a new scheme based on the frequency of the index count [25]. The frequency of each index, i.e. the number of occurrences of each index, is considered. Based on ranking frequency, the secret data is hidden as part of higher frequency indices which leads to improved embedding capacity.

Yang et al.'s scheme [26], the VQ codebook is sorted using referred counts and the codebook is divided into 2^B clusters. Here *B* denotes the size of the secret data that is embedded as part of each VQ index. The embedding capacity achieved with this scheme is better than a scheme of Chang et al.'s. This scheme proposed by Dushyant Goyal et al. [27] is a reversible data hiding technique based on VQ encoding. The embedding capacity is increased by increasing embeddable index values. The performance of this scheme in terms of embedding capacity is better than that of Yang et al.'s. In this scheme, the original VQ compressed index values can be recovered without any loss after extracting the secret data that is embedded.

This scheme proposed by Wang et al. [28] is based on Adjoining State Codebook Mapping (ASCM). This ASCM maps the contents of an image block to the index of the corresponding code vector in the Codebook. In this scheme, two state codebooks are created by using the adjacent image block of a current image block that is being encoded. So this leads to accurate representation of current image block. This leads to improved coding efficiency and increasing embedding capacity of secret data. In Lin et al.'s [9] is used two techniques of SOC and State Codebook Mapping (SCM) to reduce the VQ index size. This leads to embedded the large amount of secret data. This scheme outperforms Wang et al. in terms of embedding capacity and compression rate.

III. PROPOSED METHOD

The generated initial Codebook is optimized using K-Means Clustering technique to improve the quality of the codebook and the optimized codebook is used in the proposed methods. In this research paper, two different approaches are proposed to increase the embedding capacity and the complexity involved in decoding the secret data. In the first approach, the codebook is modified by replacing few of the elements of code vectors. In the second method, the embedding capacity is further improved by embedding secret data as part of IndexMap in addition to Codebook.

A. Steganography using Vector Quantization with Modified Codebook (SVQMCB)

The codebook consists of M number of codevectors, each of which has 16 elements {c1, c2, c3,…., c16}. The optimal size of Codebook is generally 256 as it helps maintaining a trade-off between quality and coding efficiency. A codebook therefore contains 256 x 16 elements. In each codevector, 4 elements from predetermined positions (p) computed using (3) are replaced with four 8-bit secret data and the embedding capacity is thus 256 x 4 x 8 bits.

$$p = i \bmod 4 \quad where\ p = 2 \quad and\ 1 <= i <= 16 \tag{3}$$

In the receiving end, after retrieving the embedded data, gaps will be introduced. The inter pixel feature of images is exploited and hence these gaps are filled by taking the average of the neighboring values, which will be an approximation of the original value without much distortion in the codebook. The gaps introduced in the aforementioned positions are filled up with the values generated using (4).

$$C_i = \begin{cases} (C_{i-1} + C_{i+1})/2 & if\ (i \bmod 4) = 2 \\ C_i & Otherwise \end{cases} \quad where\ 1 \le i \le 16 \tag{4}$$

B. Steganography using Vector Quantization with Extended Codebook (SVQECB)

In this approach, the actual elements of codevector are remain untouched. Eight more elements are added to the end of each codevector. Pixel values from four desired positions (computed using (3)) are selected. The absolute differences between the selected pixel values and the four 8-bit secret data elements are computed using (5). These differences are padded to the codevector as c17, c18, c19 and c20.

$$d_j = |C_i - sd| \ where\ 17 \le j \le 20 \ \ and\ i = 2,6,10,14 \tag{5}$$

Where C_i is the i[th] element of a codevector value and *sd* is the secret data to be embedded as part of codebook. d_j is the absolute difference values. The elements C_{21} to C_{24} of a codebook are entered either as1 or 0 depending on the sign of the difference. Now the manipulated codevector will have the elements as

$$CV = \{C_1, ..., C_{16}, C_{17}, C_{18}, C_{19}, C_{20}, C_{21}, C_{22}, C_{23}, C_{24}\}$$

where $C_1, C_2, \ldots, C_{16}$ are the actual pixel values of the original codevector. The elements C_{17}, C_{18}, C_{19} and C_{20} denote the absolute difference values and the remaining elements C_{21}, C_{22}, C_{23}, C_{24} are the indicator flags.

At the receiving end, while restoring the steganographed data, if flag is '1', the ci value is subtracted from the original pixel value and if the flag is '0', the ci value is added to the respective pixel value to get the secret data using equation (6). The identification of pixel values with which the data hidden is more complex when compared to that of the previous technique and also has an additional overhead of increased 32 bits for each codevector.

$$SD = \begin{cases} P_j - c_i & \text{if flag} = 1 \\ P_j + c_i & \text{Otherwise} \end{cases} \quad \text{where i} = 17, 18, 19, 20 \tag{6}$$

where P_j is the pixel value and c_i is the absolute difference.

B.1 Algorithm

Encoding:

Step 1: Generate Codebook (CB) and refine using K-Means Clustering Algorithm.
Step 2: Generate IndexMap (IM).
Step 3: For each Codevector (CV) in CB, perform the following:
- **Step 3.1:** Identify the four-pixel positions in CV.
- **Step 3.2:** Compute the absolute difference between the selected pixels and four bytes of secret data.
- **Step 3.3:** Generate a flag sequence F depending on the nature of difference values.
- **Step 3.4:** Pad the codevector with the set of absolute differences and Indicator Flags.

Step 4: Store or transmit the modified CB and IM.

Decoding:

Step 1: For each codevector (CV) in CB, perform the following to retrieve the secret data back:
Step 2: Extract secret data using the entire codevector using (6).
Step 3: Retain the first 16 values for Image reconstruction.

C. Data Embedding in IndexMap using SOC

In this method, a novel idea of embedding the secret data in Index Map using Search Order Coding (SOC) Technique is introduced. When implementing SOC algorithm, the first row and first column of indices do not undergo any process as they do not have left and top neighbors. The remaining indices are read in Raster Scan Order. The index that is currently read is called Search Center (I_c) and is compared against the neighboring horizontal and vertical indices I_l, I_{tl}, I_t and I_{tr}, where l, lt, t and tr indicate Left, Top Left, Top and Top Right indices. These Search indices are assigned the Search Order Codes 00, 01, 10 and 11 respectively. If I_c matches any of the neighboring indices of the Search-path, a bit sequence (of size 9) is generated by combining the following bits: 1 bit (1 - an indicator bit that indicates if there is a matching neighbor), corresponding search order code of size 2 bits and the remaining 6 bits that represent the secret data to be hidden. Now, the original index is replaced with the new bit sequence that is generated. If there is no match between the Search Centre and its neighboring pixels, the Index I_c (generally 8 bits long) is just prefixed with the indicator bit '0'.

Index Map

18	56 (01)	96 (10)	81 (11)
108	38 (00)	**56**	92
83	91	38	56
114	83	108	117

→ I_c

Fig. 1. Search Order Coding

In Fig. 1, the highlighted cell is the Search Center and is equal to the top-left index. This index is encoded as 101xxxxxx, where the first bit is the indicator bit, the next two bits represent search order code (01) and the remaining 6 bits represent the data to be hidden. This methodology leads to increased complexity in identifying the data.

C.1. Algorithm

Encoding

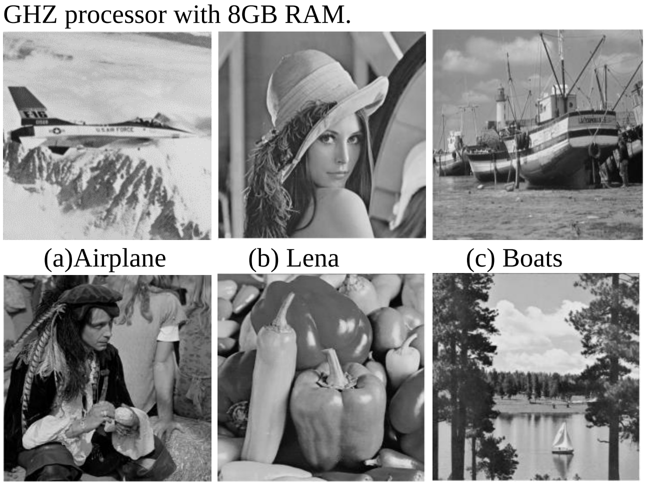
Step 1: Generate the IndexMap (IM) by encoding the cover image I using a VQ Encoder.
Step 2: Read the VQ index Ic from the IndexMap 'IM' that forms the Search Center.
Step 3: If I_c matches with any of SOCs, generate bit sequence using an indicator flag '1' followed by the respective SOC and then concatenate with 6 bits of secret data. Replace I_c with the generated bit sequence.
Step 4: If no match found, prefix Ic with the indicator flag '0'.
Step 5: Perform the steps 3 & 4 for all the indices one by one.

Decoding

Step 1: For each index in the IndexMap, perform the following:
Step 2: if the first bit is 0, the index I_c does not contain any secret data. Go to Step 5.
Step 3: Else, the 2nd and 3rd bits are used to retrieve the Index from the respective position.
Step 4: Get the secret data from the last six bits.
Step 5: Repeat the above steps till all the indices of IndexMap are processed.

IV. EXPERIMENTAL RESULT AND DISCUSSIONS

All experiments are performed with the benchmark gray scale images of size 512 x 512 pixels. The input images are Airplane, Lena, Boats, Man, Peppers, Sailboat, Zelda, Girl and Couple. Fig. 2 shows the input images taken for the study. The algorithms are implemented using MATLAB R2014b. The Hardware used is, the Intel® Pentium® 1.90 GHZ processor with 8GB RAM.



(a)Airplane (b) Lena (c) Boats
(d) Man (e) Peppers (f) SailBoat

(g) Zelda (h) Girl (i) Couple

Fig. 2. Input Images taken

Embedding Capacity and Visual Quality

While embedding the data, in the first method, the original pixel values of the codevectors are replaced with the secret data and the pixel values are modified during decryption leading to acceptable degradation in image quality. In other two methods, lossless strategy is adopted to retain the quality of codebook. The Peak Signal-to-Noise Ratio (PSNR) is used to measure the image quality. PSNR is defined as:

$$PSNR = 10\log_{10}\left[\frac{255^2}{MSE}\right] \qquad (7)$$

where, MSE is the Mean Square Error between the original image and stego image. MSE is defined as:

$$MSE = \frac{1}{MxN}\sum_{i=1}^{M}\sum_{J=1}^{N}(I(i,j) - I'(i,j))^2 \qquad (8)$$

where, *M* is the number of rows and *N* is number of columns. *I(i, j)* and *I'(i, j)* denote the Original image and Stego-image respectively.

The bit-rate is used to analyze the coding efficiency of any Compression algorithm. The bitrate is computed using (9), where C represents the size of the compressed image in terms of bits. M x N is the size of the original image in terms of rows and columns. The bitrate obtained is represented in terms of bits per pixel (bpp). The lower the bitrate, higher the coding efficiency is.

$$bitrate = \frac{C}{MxN} bpp \qquad (9)$$

The PSNR values, bitrate and embedding capacity achieved with the proposed methods are shown in Table I. In SVQMCB, average PSNR and bitrate obtained are **34.38** and **0.69** respectively. In SVQECB, the average PSNR and bitrate are **34.46** and **0.72** respectively. In the existing method, the secret data has been embedded only as part of IndexMap. But in our proposed methods, the data has been embedded as part of both Codebook and IndexMap that leads to increased embedding capacity. On an average, **8192** bits of secret data are embedded in Codebook and **39063** bits of data in IndexMap, which is a significant improvement when compared to that of similar existing techniques. Hence the total average embedding capacity obtained with our proposed method is **47255** bits. For visual comparison, the reconstructed images of our proposed methods are given in Fig. 3 and Fig. 4.

Table I: Comparison of PSNR, Bitrate and Embedding Capacity with respect to the proposed Methods: SVQMCB and SVQECB

Images	SCG [22]		Proposed Methods				Embedding Capacity		
			SVQMCB		SVQECB				
	PSNR	***BPP***	***PSNR***	***BPP***	***PSNR***	***BPP***	***CB***	***Index Map***	***Total***
Airplane	35.75	0.63	35.59	0.69	35.75	0.72	8192	47472	55664
Lena	36.12	0.63	36.03	0.69	36.12	0.72	8192	45912	54104
Boat	34.16	0.63	34.01	0.69	34.16	0.72	8192	36984	45176
Man	33.38	0.63	33.35	0.69	33.38	0.72	8192	33684	41876
Peppers	35.19	0.63	35.10	0.69	35.19	0.72	8192	56712	64904
SailBoat	33.51	0.63	33.39	0.69	33.51	0.72	8192	30186	38378
Zelda	37.74	0.63	37.73	0.69	37.74	0.72	8192	37500	45692
Girl	36.87	0.63	36.85	0.69	36.87	0.72	8192	50184	58376
Couple	33.73	0.63	33.55	0.69	33.73	0.72	8192	42930	51122
Baboon	31.24	0.63	31.21	0.69	31.24	0.72	8192	22452	30644
Average	**34.77**	**0.63**	**34.68**	**0.69**	**34.77**	**0.72**	**8192**	**40402**	**48594**

The bitrate and the embedding capacity obtained with the proposed methods are compared with that of the results obtained with the existing techniques proposed by Chang et. al., Yang et. al. and Dushyant Goyal et. al. in Table II.

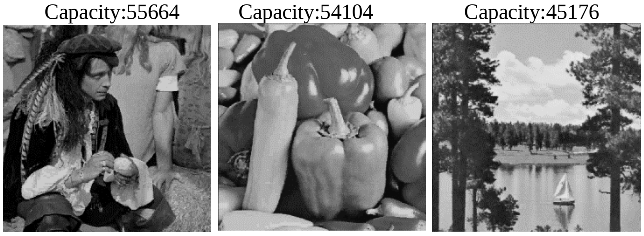
Table II: Comparison of Proposed scheme and the existing techniques with respect to Bitrate and Embedding Capacity (EC)

Images	Chang et al.[24]		Yang et al.[26]		Dushyant Goyal et al.[27]		Our Proposed	
	BPP	Embedding Capacity	BPP	Embedding Capacity	BPP	Embedding Capacity	BPP	Embeddin g Capacity
Lena	0.63	16003	0.63	26980	0.70	30559	0.72	54104
Airplane	0.63	16003	0.60	28840	0.64	32701	0.72	55664
Peppers	0.60	16003	0.61	28818	0.66	32521	0.72	64904
Zelda	0.65	16003	0.65	27304	0.69	29679	0.72	45692
Average	**0.63**	**16003**	**0.62**	**27986**	**0.67**	**31365**	**0.72**	**55091**

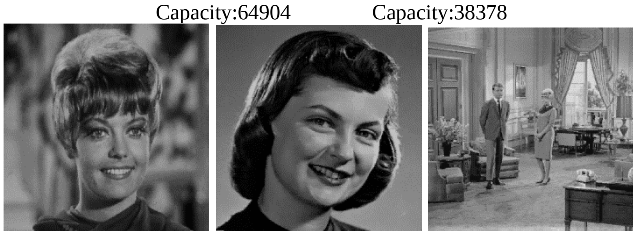
Retrieval Number: E2547039520/2020©BEIESP
DOI: 10.35940/ijitee.E2547.039520
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

PSNR:35.59 BPP:0.69 Capacity:55664 | PSNR:36.03 BPP:0.69 Capacity:54104 | PSNR:34.01 BPP:0.69 Capacity:45176



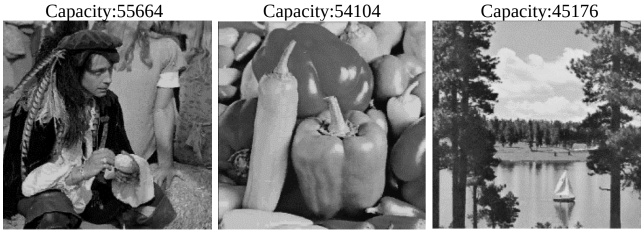
PSNR: 33.35 PSNR:35.10 PSNR:33.39
BPP:0.69 BPP: 0.69 BPP:0.69 Capacity:41876
Capacity:64904 Capacity:38378



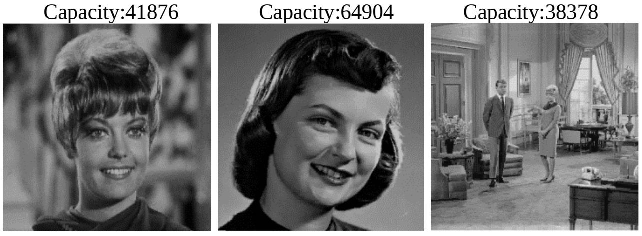
PSNR:37.73 BPP:0.69 Capacity:45692 | PSNR:36.85 BPP:0.69 Capacity:58376 | PSNR:33.55 BPP:0.69 Capacity:51122

Fig. 3. Reconstructed images after extracting the secret data using SVQMCB (proposed) method

PSNR:35.75 BPP:0.72 Capacity:55664 | PSNR:36.12 BPP:0.72 Capacity:54104 | PSNR:34.16 BPP:0.72 Capacity:45176



PSNR:33.38 BPP:0.72 Capacity:41876 | PSNR:35.19 BPP: 0.72 Capacity:64904 | PSNR:33.51 BPP: 0.72 Capacity:38378



PSNR:37.74 BPP: 0.72 Capacity:45692 | PSNR:36.87 BPP: 0.72 Capacity:58376 | PSNR:33.73 BPP: 0.72 Capacity:51122

Fig. 4. Reconstructed images after extracting the secret data using SVQECB (proposed) method

From Table 2, it is observed that the Embedding capacity has been increased significantly by the proposed method, though there is slight increase in the bitrate of the compressed image.

On an average, Chang et al. [24] achieved an embedding capacity of **16,003** bits (uniform for all images) and the proposed method (adaptive) leads to an embedding capacity of **55091** bits, which is 244.25% more. The Yang et al. [26] achieved an embedding capacity of **27986** bits and the proposed method leads to an embedding capacity of **55091** bits, which is 96.53% more. The DushyantGoyal et al. [27] achieved an embedding capacity of **31365** bits and the proposed method leads to an embedding capacity of **55091** bits, which is 75.20% more than that of DushyantGoyal et al. There is a significant improvement in Embedding capacity of the proposed method. The performance efficiency of the proposed method in terms of Embedding Capacity is represented graphically in Fig. 5.

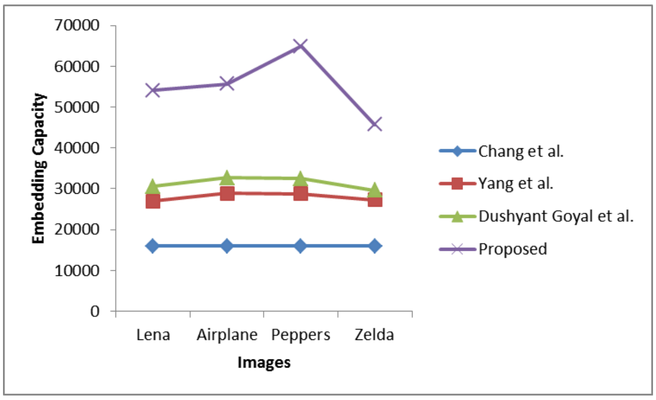


Fig. 5. Graphical Representation of Performance efficiency in terms of Embedding Capacity of the proposed method.

Table III. Comparison of experimental results of our proposed scheme with Wang et al. and Lin et al. in terms of Embedding Capacity

Images	Embedding Capacity		
	Wang et al.[28]	**Lin et al.[9]**	**Proposed Method**
Lena	44101	48096	54104
F16	37781	47456	55664
Baboon	9305	12422	**34731**
Peppers	34721	44985	**64904**
Boat	23181	29276	43824
Girl	22106	35067	51314
Toys	49201	70477	56697
Tiffany	34991	53048	52476
Average	**31923**	**42603**	**51714**

Table III shows that the maximum embedding capacity achieved with the proposed method is 64904 bits with ‘Peppers’ image and the minimum capacity is 34731 bits with ‘Baboon’ image. Wang et al. [28] and Lin et al.’s [9] have come out with 9305 bits and 12422 bits respectively for the Baboon image. For Toys image, the proposed method has achieved with the embedding capacity of 56697 bits, which is 7496 bits more than that of Wang et al.’s (49201 bits), but slightly less than that of Lin et al.’s (70477 bits).

In an average, the embedding capacity achieved with the proposed method is 51714 bits, which is better than that of Wang et al.’s (31923 bits) and Lin et al.’s (42603 bits). The results are graphically compared in Fig. 6**.**

Retrieval Number: E2547039520/2020©BEIESP
DOI: 10.35940/ijitee.E2547.039520
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

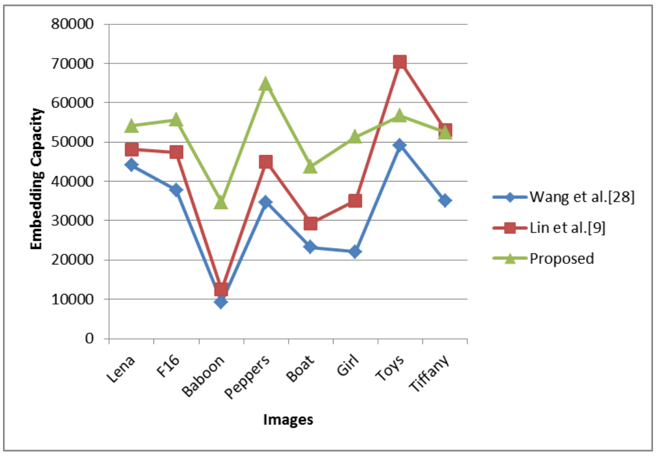


Fig. 6. Embedding Capacity of Proposed Method with Wang et al. and Lin et al. scheme

V. CONCLUSION

This paper proposes both reversible and irreversible data hiding schemes based on Vector Quantization. In the first method, being irreversible, secret data is embedded as part of the codevector. Pixels in any four predetermined positions are used to embed secret data. In extracting phase, the replaced pixel values are interpolated by taking the average of neighboring values. This method is irreversible. In the second method, that is reversible, the size of the codebook is increased by padding the difference values and flag values as part of the codevectors that leads to more complexity in getting back the secret data. The difference and flag values are used to recover the data from the compressed data. The usage of SOC as part of IndexMap improves the coding efficiency of VQ and also leads to more embedding complexity. It has been observed from the results that the proposed method outperforms the existing steganography methods in terms of embedding capacity. In an average, the embedding capacity achieved with the proposed method is 51714 bits, which is better than that of Chang et al. (16003 bits), Yang et al. (27986 bits), DushyantGoyal et al. (31365 bits), Wang et al. (31923 bits) and Lin et al. (42603 bits). In our proposed method, the bitrate is little higher than the existing method. In future, the proposed work may be improved to reduce the bitrate without affecting the quality of the Cover Image and to improve the embedding capacity.

REFERENCES

1. Chin-Chen Chang, Wei-Liang Tai & Chia-Chen Lin, "A Reversible Data Hiding Scheme Based on Side Match Vector Quantization", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 16, No. 10, 2006.
2. R.M.Gray, "Vector Quantization", IEEE Acoustic, Speech, Signal Processing., Vol. 1, 1984, pp. 4-29.
3. T.Kim, (1992) "Side Match and Overlap Match Vector Quantizers for images", IEEE Transactions on Image Processing, Vol. 1, No. 4, 1992, pp. 170-185.
4. W.B.Pannebaker & J. L. Mitchell, "The JPEG Still Image Data Compression Standard", New York: Reinhold, 1993.
5. D.S.Taubman & M. W. Marcelin, "JPEG2000: Image Compression Fundamentals Standards and Practice", Norwell, MA: Kluwer, 2002.
6. Wei-Jen Wang, Cheng-Ta Huang & Shiuh-Jeng Wang, "VQ Applications in Steganographic Data Hiding upon Multimedia Images", IEEE System Journal, Vol.5, No.4, 2004.
7. Chin-Chen Chang, Thai Son Nguyen & Chia-Chen Lin, "A Novel VQ based Reversible Data Hiding Scheme by Using Hybrid Encoding Strategies", Journal of System and Software, Vol.86, 2013, pp. 389-402.
8. Po-Hsin Chang, Ko-Chin Chang & Chiang-Lung Liu, "A reversible Data Hiding Scheme for VQ Indices using Histogram Shifting of Prediction Error", Multimedia Tools and Applications, 2015.
9. Chia- Chen Lin, Xiao-Long Liu & Shyan Ming Yuan, "Reversible Data Hiding for VQ Compressed Images based on Search-Order Coding and State-Codebook Mapping", Information Sciences, Vol.293, 2015, pp.314-326.
10. Peyman Rahmani & Gholamhossein Dastghaibyfard, "An Efficient Histogram-Based Index Mapping Mechanism for Reversible Data Hiding in VQ Compressed Images", Information Sciences, Vol.435, 2018, pp.224-239.
11. Zhibin Pan & Lingfei Wang, "Novel Reversible Data Hiding Scheme for Two Stage VQ Compressed Images based on Search-Order Coding", Journal of visual communication and Image Representation, Vol. 50, 2018, pp.186-198.
12. Ling fei Wang, Zhibin Pan & Ruoxin Zhu, "A Novel Reversible Data Hiding Scheme by Introducing Current State Codebook and Prediction Strategy for Joint Neighboring Coding", Multimedia Tools and Applications, Vol.76, No. 24, 2017, pp. 26153-26176.
13. Ling fei Wang, Zhibin Pan & Ruoxin Zhu, "A Novel Reversible Data Hiding Scheme using SMVQ Prediction Index and Multi-Layer Embedding", Multimedia Tools and Applications, Vol.76, No.24, 2017, pp. 26225-26248.
14. Binbin Xia, Anhong Wang, Chin-Chen Chang & Li Liu, "Reversible Data Hiding for VQ Indices using Hierarchical State Codebook Mapping", Multimedia Tools and Applications, Vol.77, No.16, 2018, pp. 20519-20533.
15. Kris Manohar & The DucKieu, "A Centroid based Vector Quantization Reversible Data Hiding Technique", Multimedia Tools and Applications, 2019, pp. 1-26.
16. Y.Linde, A.Buzo & R.M.Gray, "An Algorithm for Vector Quantizer Design", IEEE Transactions on Communications, Vol.28, No.1, 1980, pp.84-95.
17. S.Vimala, (2011) "Convergence Analysis of Codebook Generation Techniques for Vector Quantization using K-Means Clustering Technique", International Journal of Computer Applications, Vol.21, N0.8.
18. K.Somasundram & S.Vimala, "A Novel Codebook initialization Technique for Generalized Lloyd Algorithm using Cluster density", International Journal on Computer Science and Engineering, Vol. 2, No. 05, 2011, pp. 1807-1809.
19. K.Somasundram & S.Vimala, "Fast Codebook Generation for Vector Quantization using Ordered Pairwise Nearest Neighbor with Multiple Merging", IEEE Conference (ICETECT), 2011.
20. K.Somasundram & S.Vimala, "Codebook Generation for Vector Quantization by sorting of Sum Vectors", CiiT International Journal on Digital Image Processing, Vol. 2, No. 8, 2010.
21. K.Somasundram & S.Vimala, "Fast Encoding Algorithm for Vector Quantization", International Journal of Engineering Science & Technology, Vol. 2, No. 9, pp. 4876-4879.
22. K.Somasundram & S.Vimala, "Simple and fast Ordered Codebook for Vector Quantization", Proceedings of National Conference on Image Processing (NCIMP), 2010.
23. C.H. Hsieh & J.C. Tsai, "Lossless Compression of VQ index with Search Order Coding", IEEE Transactions on Image Processing, Vol. 5, No.11, 1996, pp. 1579-1592.
24. C.C. Chang, G.M. Chen, & M.H. Lin, "Information hiding based on search-order coding for VQ indices", Pattern Recognition Letters, Vol.25, 2004, pp.1253-1261.
25. C.C. Chang, W.C. Wu & Y.C. Hu, "Lossless Recovery of a VQ Index Table with Embedded Secret Data", Journal of Visual Communication and Image Representation, Vol. 18, No. 3, 2007, pp. 207-216.
26. C.H. Yang & Y.C. Lin, "Reversible Data Hiding of a VQ Index Table based on Referred Counts", Visual Communication and Image Representation, Vol. 20, No. 6, 2009, pp. 399-407.
27. Chin-Feng Lee, Kuan-Ting Li, & DushyantGoyal,"High Capacity Lossless VQ hiding with Grouping Strategy", Recent Progress in Data Engineering and Internet Technology, Vol. 1, 2011, pp. 203-209.

AUTHORS PROFILE

P. Uma, pursued her UG Degree in M.K.University, P.G. Degree in Anna University, M.Phil. Degree in Mother Teresa Women's University. She is doing her research in Mother Teresa Women's University. She has authored 10 research papers in reputed International Journals. Her research focus is on Image Compression and Steganography.

S. Vimala, pursued her UG Degree in M.K.University, P.G. Degree in Bharathiar University and M.Phil. & Ph.D. Degrees in Mother Teresa Women's University. She has authored nearly 40 research papers in reputed international journals and a book on 'C Programming'. She is a life member of IACSIT. She has been awareded the 'Best Thesis Award' by ICFAI Business School, Hyderabad and Michigan State University, USA. Her research focus is on Image Compression, Image Analysis and Cloud Computing. She has twenty years of teaching experience and 10 years of research experience.