Grenze International Journal of Engineering and Technology, July 2017

Implementation of Gap Algorithm and Adaptive Arithmetic Coding for Loseless Image Processing

Dilip kumar.V[1] and Murugesh. K[2]
[1]M. Tech Student, Department of Electronics & Communication Engineering, Don Bosco Institute of Technology, Bengaluru, Karnataka, India
[2]Research Scholar, Department of Electronics & Communication Engineering, Don Bosco Institute of Technology, Bengaluru, Karnataka, India

***Abstract*—Digital images are used in different applications like aerospace, military, science and many more applications. The image compression is important in data storage and data transmission through an untrusted channel to make use of efficient network bandwidth. In most of the case image compression is done by network provider. This leads to data loss while compressing, here we will not able to get exact image even though it is lossless compression if we encrypt and decrypt. To overcome this problem, we have designed an efficient image encryption and compression. The image compression is done by transmitter after the encryption, which gives lossless image at the receiver side. To encrypt the image, the prediction error and random permutation techniques were used and the GAP algorithm is used to find the prediction error. Adaptive Arithmetic coding algorithm is used to compress the encrypted image for lossless image. In this work, the encrypted and compressed image is decompressed and decrypted without any loss of data at the receiver side.**

***Index Terms*— Compression, GAP algorithm and Adaptive Arithmetic coding.**

I. INTRODUCTION

In rapid development of transmission and network technologies, the safety of transmission becomes a lot of necessary, since transmission information are transmitted over open networks. Decreasing the extent of an image with or without a loss of some information is called Image compression. This is required for reducing the data transfer capacity of the image so as to facilitate easy transmission over a channel of smaller capacity.
For security data transmitted over the channel, a technique called “Encryption” is used. This is the method for shielding data from hackers. It changes the data into a structure such that information cannot be accessed by hackers. Encryption basically is a method of scrambling the data, or hiding the data or water marking to overlap the data or some other masking of the data. The data may be a text, an image, an audio or a video. The data “decryption and de-compression” techniques are used to retrieve back the information from the encrypted & compressed data.

In most of the practical scenarios, usually when the content owner transmits confidential data, the content owner wishes to safely and efficiently transmit to the receiver. In majority cases this will be done through a channel that cannot be trusted. The content owner (Transmitter) initially compresses I_c the data using a suitable technique and then encrypts the compressed data through an appropriate encryption algorithm. The

Grenze ID: 01.GIJET.3.3.95

resulting encrypted I_{ce} data is then passed all the way to the channel provider who simply forwards it to the receiver section [1].

At the receiver end, the decryption and decompression operations are respectively performed to recover the original data as shown in the Fig 1.1. Compression-then-Encryption (CTE) concept fulfils the prerequisites of a secure transmission system. However, compression is expensive and it is usually done by the network provider. Hence it is sufficient for the content owner to do encryption and send the encrypted data to the service provider. Now, since the service provider does not have the key, the data will indeed be secure. There are some more drawbacks of the traditional system as explained next [2].

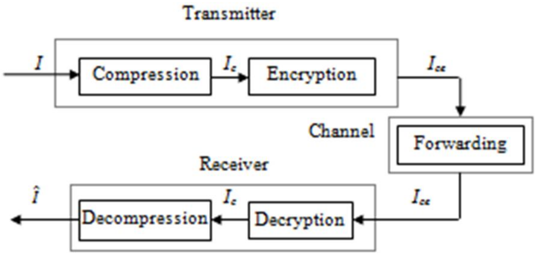


Fig. 1.1: Traditional Compression-then-Encryption (CTE) System

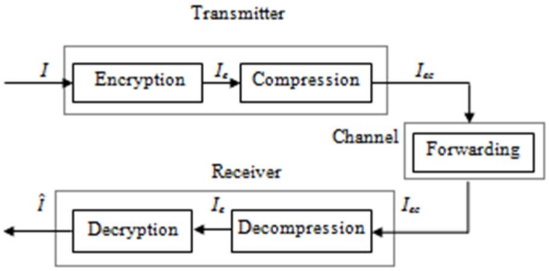


Fig. 1.2: Encryption-then-Compression (ETC)

When it comes to the transmission of images, the quality depends on the compression ratio. Further, decompression will be applied to the decrypted image. As a result, the losses in the decryption phase will add to those of decompression and thus the traditional system does not appear that attractive for communication.To overcome above drawback in this paper we designed Encryption-then Compression (ETC) frameworks. Here encryption is done before the compression. This will give more security and lossless compression of the image Fig 1.2 illustrate ETC framework.

II. Proposed Work

The proposed work has three phases like: Encoding of image, Compression, Decryption, and cluster is that the new methodology used for image encoding and compression. The phases are:

A. Image Encryption

The first stage in the proposed work is image encryption. Here the source image is divided in to pixels. For each pixel a prediction value is prepared using an appropriate image predictor. A mapping is done by calculating the prediction errors. The image predictor used in our project is "Gradient Adjusted Predictor" (GAP). This predictor converts the applied image into pixels and assigns appropriate values. These pixels are grouped together to form "Clusters". A random permutation technique is applied to each cluster so as to scramble the pixels. At the assembler all the scrambled clusters are combined to get back the encrypted image. The encryption phase is shown in Fig 2.1 [3].

The recursive procedure of playing the image encoding is then given as follows:

Step 1: Determine, for the whole image, all the prediction errors.

Step 2: Split or divide the prediction errors so computed into L clusters. Each cluster is formed by the concatenation of the mapped prediction errors in a raster- scan order.

Step 3: Reshape the prediction error in each cluster into a 2-D block each block will have four columns and $\{|C_k|/4\}$ rows. Here $|C_k|$ represents the number of prediction errors in the cluster C_k, $0 \leq k \leq (L\text{-}1)$.

Step 4: To each of the resulting prediction error block, perform two key- driven cyclical shift operations and read out the data in raster- scan order. This will then, result in permuted cluster,$\tilde{C}_k$. Let CS_k be the secrete key vector controlling the column offset of C_k and RS_k represent the secret key vector controlling the row offset of $C_{k.}$ These vectors are obtained from the key stream generated by a stream cipher. This means that the key vector used here may be different in other words; the same image encrypted at different time instant may have different key vector indicating randomness. Fig 2.2 illustrate the random permutation for an input sequence $S = s_1\ s_2 \ldots s_{16}$. The subscripts indicate the indices of the element of S. Prior to permutation, the first row will be [1, 2, 3, 4], the second row will be [5, 6, 7, 8], and so on. The key vector CS = [2 3 0 1]

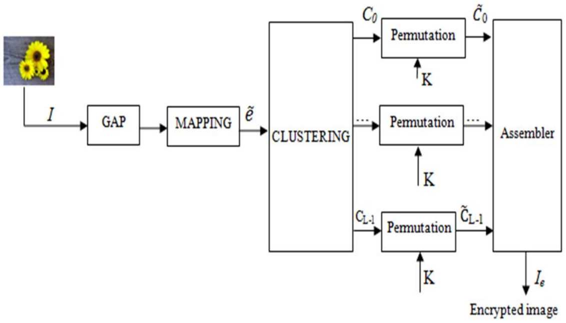


Fig. 2.1: Encryption phase

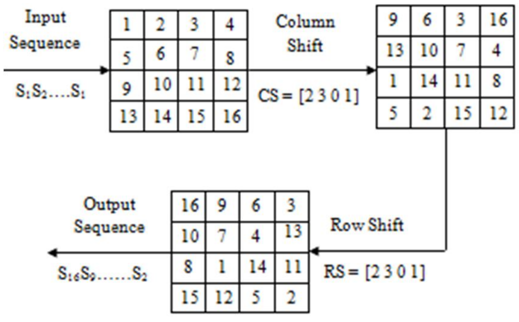


Fig.2.2: An example of the cyclical shifts

specifies the column shifts. Each column will undergo a downward cyclic shift as per the associated key value of that column. Similar procedure is repeated for each of the rows by using another key vector RS = [1 3 1 2]. Observe that these permutation operations are realized in circular shifts. Hence they may be implemented usually in either software or hardware.

Step 5: All permuted clusters C_k, $0 \leq k \leq L-1$, obtained in step 4 are passed on to the assembler for concatenation and generation of the final encrypted image.

$I_e = \tilde{C}_0 \tilde{C}_1 \ldots . \tilde{C}_{L-1}$…………………………………… (2.1)

Step 6: The encrypted image Ie together with the length of each cluster $|\tilde{C}_k|$, $0 \leq k \leq L-2$, is passed on to the compression unit / forwarded to the network provider/ channel.

B. Gradient Adjusted Predictor (GAP)

Gradient adjusted predictor (GAP) is an edge detection algorithm. It is based on gradient estimations around the current pixels (Context based entropy coding). The GAP is a simple, adaptive nonlinear predictor. It can adapt itself to the intensity gradient in the vicinity of the predicted pixel. GAP distinguishes 3 kinds of edges – strong edge, straightforward edge and soft edge. GAP has high flexibility to totally different regions. A context model as shown in Fig 2.3 is obtained be assigning n, w, ne, nw, nn, ww, nne denote north, west, north-east, north-west, north-north, west-west and north-north-east abbreviations to each pixel X [i, j]. If there is a reasonable likelihood that vertical or horizontal edge exists, then predictor becomes pixel n or pixel w. otherwise the pixel is selected from different choices depending on the local smoothness [4] - [5].

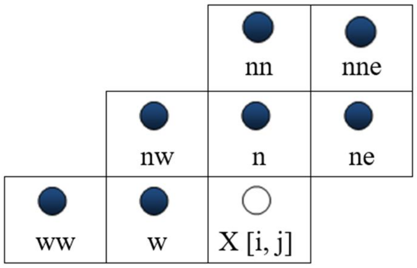


Fig. 2.3: The GAP predictor

C. Compression Phase

Compression is achieved by encoding there are different methods available: Adaptive arithmetic coding Run length coding Huffman coding. In this paper k Adaptive arithmetic coding is using to compress the encrypted image. Adaptive arithmetic coding is applied to each cluster of the image obtained in step 6 of the encryption phase. The actual procedure is illustrated in Fig 2.4.

The encrypted image (bit stream) is de assembled to get back the cluster Adaptive arithmetic coding is applied each of the cluster the encoded clusters are re assembled to produce the final compressed image. Basically Adaptive arithmetic coding a maps a string of data in to a string of code word in such a manner that original data can be recovered by de the decoding operation. One recursion of the arithmetic coding algorithm handles one de assembled cluster at a time [6] - [7].

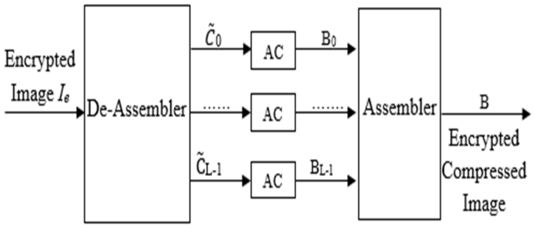


Fig. 2.4: Compression Phase

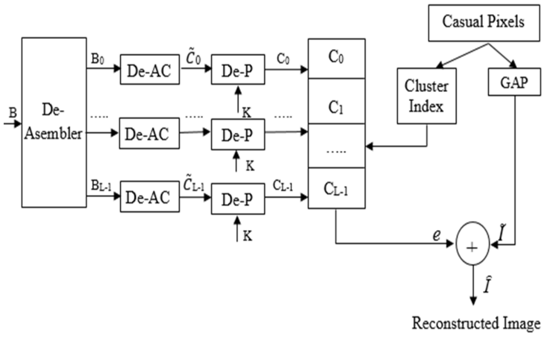


Fig. 2.5: Decryption phase

D. *Decryption Phase*

This phase is the reverse operation of encryption and compression phase. Here the receiver will recover the encrypted image by the decompression operation using an appropriate key. To be specific, the compressed bit stream received by the receiver will be dissembled back into encoded cluster Adaptive arithmetic decoding algorithm is applied to each encoded cluster after this reverse operation of row wise cyclic shift and column wise cyclic shift are performed using suitable key which were used in encryption phase. After this step the cluster prediction error are assembled and added with the pixels (Prediction). The different stages of image reconstruction are shown in Fig 2.5 [8].

III. PERFORMANCE MEASURES

To study the relative performance of cluster primarily based segmentation strategies the subsequent quality measures square measure calculated.

A. Security Analysis

The security analysis of the proposed technique is done by using the comparative analysis supported by appropriate performance factors. As the proposed methodology involves encryption, compression and de-compression-decryption, a three phase strategy the analysis of performance will be done on the following criteria.

B. Visual Analysis

Observation is an important factor for testing an encrypted image. A good encryption algorithm should not revile the identity and features of the image to any hacker in other words information shall not be visually detectable. Since the image is encrypted first and then compress, this aspect of security is fulfilled. The hacker should know the decompression key and decryption key as well as the order of encryption and compression operations. Specifically, the hacker will not know whether the encryption is done first or compression is done first. Thus visual safety of the image is achieved safely.

C. Mean Square Error (MSE)

Mean square error is the average squared difference between the pixel values of the original image and the encrypted image. MSE should be larger for a better performance. If it is too small, a hacker will be able to accesses the image easily. The MSE is computed as below:

$$MSE = \frac{\sum_{i,j}^{M,N}[I(i,j) - I_e(i,j)]^2}{M * N} \dots\dots\dots\dots\dots\dots\dots..\dots (2.2)$$

M*N represent the size of the image. Where I (i, j) is original image and Ie (i, j) is encrypted image.

D. PSNR

The encryption quality is reflected by the Peak signal-to-noise ratio. A good encrypted image will have a low value of PSNR. It is computed using the formula:

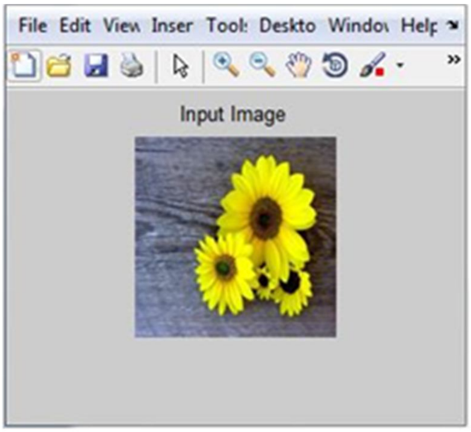
$$PSNR = 10 \log\left(\frac{256^2}{MES}\right) \quad dB \ldots\ldots\ldots\ldots\ldots\ldots\ldots . (2.3)$$

IV. RESULTS

In this section the analysis on the performance of encoding then compression system is shown. This section explains that tools square measure utilised and also the overall flow of the project is explained very well.

A. At transmitter Side

The image in the Fig 4.1 is the desired input image for encryption then compression.The dimensions of the image is 128*128.



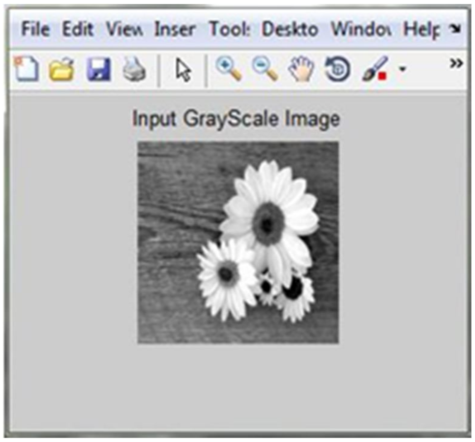
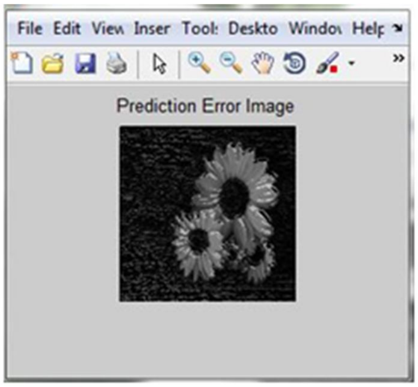


Fig.4.1: Input image

Fig.4.2: Gray scale image

MATLAB is most apt for Gray-Scale image processing rather than colour image as it has to operate in 3 different co-ordinates i.e. RGB planes. Hence in this paper the images are processed in gray scale for which it is converted from colour to gray scale image as shown in Fig 4.2. After converting the colour image to gray scale image, the above Fig 4.3 shows the mapped prediction error image which is formed by the difference of original image and the prediction made by the GAP image predictor Prediction error image which is divided into eight clusters and each and every cluster is created by concatenating the mapped prediction errors.



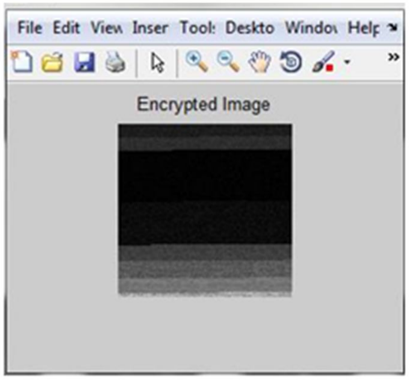


Fig. 4.3: Prediction error image

Fig. 4.4: Encrypted image

The Fig 4.4 shows the encrypted image which is formed by performing two key driven cyclical shift operations i.e. column wise cyclic shift and the row wise cyclic shift to each and every resulting prediction error block and subsequently assembling the cyclic shifted clusters and this will generates the final encrypted image. The result of prediction error clustering and random permutation will be the transmitted image as shown above. This image travels through the noisy channel amidst number of intruders. If the eavesdropper accesses the image he will retrieve the above image due to which he is unable to learn useful information regarding the original image. In this way guaranteeing the security of the secret image. This completes the encryption phase the next phase is the compression phase where we have to de-assemble to clusters and then arithmetic coding is applied and this generates the compressed bit stream.

B. At the receiver: (Decompression and decryption phase)

Once the compressed bit stream is received at the receiver section the decompression and decryption phase begins. The figure 4.5 shown below is the decompressed image which is formed by de-assembling the encrypted bit streams.

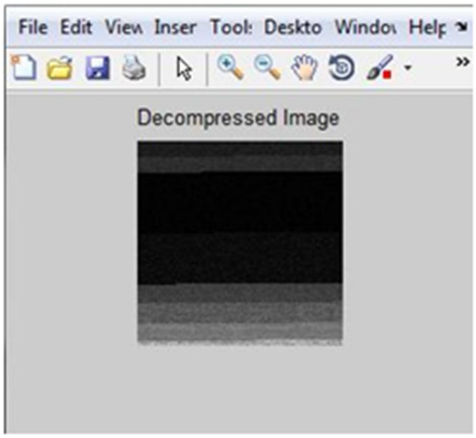


Fig. 4.5: Decompressed image

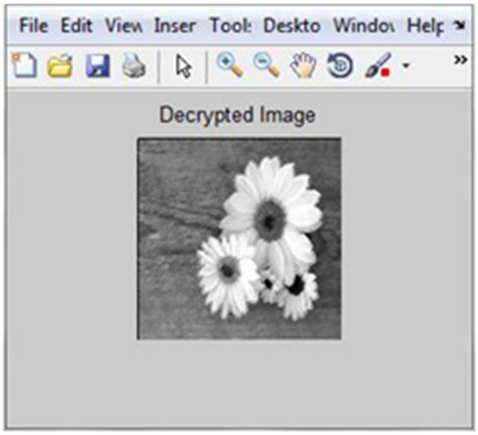


Fig. 4.6: Decrypted image

TABLE I. QUALITY MEASURES OF IMAGE

Image		Flower
Quality measures	PSNR	MSE
Prediction error image	6.6573	9.7675e+003
Encrypted image	4.4654	1.4562e+004
Reconstructed image	19.7406	3.2940e+003

In the Fig 4.6 shown is the decrypted image formed by assembling the clustered prediction errors and then adding prediction pixels with prediction errors. In the figure 4.7 shown is the final reconstructed image formed by transforming gray scale image into RGB image.

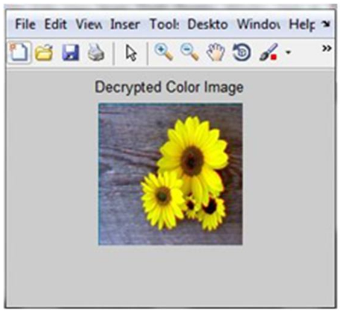


Fig. 4.7: Decrypted colour image

V. CONCLUSIONS

An ‘Encrypt- then- Compress’ (ETC) framework has been presented in this dissertation. The major concern was on security. Adam sends information to bob through network provider Charlie, who does the compression work. If the information is send without encryption it is possible that Charlie may leak it. The ETC framework presented to overcome this situation. Compressed encrypted massage requires additional key for information retrieval as compared to encrypted compressed image the former is less lossy. In this dissertation the ETC framework is implemented using GAP algorithm, AAC algorithm and their inverse counterpart. The implementation is done for a lossless image. The extension to a lossy image is straight forward. The performance analysis – based on quality measures like MSE and PSNR - reveal the efficiency of the proposed ETC frame work.

REFERENCES

[1] Shaima A. El-said, Khalid F. A. Hussein, Mohamed M. Fouad, “Securing Image Transmission Using In-Compression Encryption”

[2] R. Lazzeretti and M. Barni, “Lossless compression of encrypted greylevel and color images,” in Proc. 16th Eur. Signal Process. Conf., Aug. 2008, pp. 1–5.

[3] J. Zhou, X. Liu, and O. C. Au, “On the design of an efficient encryption then- compression system,” in Proc. ICASSP, 2013, pp. 2872–2876.

[4] X. Zhang, G. Feng, Y. Ren, and Z. Qian, “Scalable coding of encrypted images,” ……IEEE Trans. Image Process., vol. 21, no. 6, pp. 3108–3114, Jun. 2012.

[5] X. Wu and N. Memon, “Context-based, adaptive, lossless image codec,” IEEE ……Trans. Commun., vol. 45, no. 4, pp. 437–444, Apr. 1997.

[6] M. J. Weinberger, G. Seroussi, and G. Sapiro, “The LOCO-I lossless image …....compression algorithm: Principles and standardization into JPEG-LS,” IEEE Trans. Image. Process, vol. 9, no. 8, pp. 1309–1324, Aug. 2000

[7] X. Zhang, Y. L. Ren, G. R. Feng, and Z. X. Qian, “Compressing encrypted image using compressive sensing,” in Proc. IEEE 7th IIHMSP, Oct. 2011, pp. 222–225.

[8] X. Zhang, G. Sun, L. Shen, and C. Qin, “Compression of encrypted images with multilayer decomposition,” Multimedia. Tools Appl., vol. 78, no. 3, pp. 1–13, Feb. 2013.