254 to 255 during the embedding process. These created boundary pixels in the embed-ding process are defined as pseudo-boundary pixels. Boundary map is introduced to tell whether boundary pixels in marked image are natural or pseudo in extracting process. It is a binary sequence with bit "0" for natural boundary pixel, bit "1" for pseudo-boundary pixel. Later estimating errors of marginal area of $\mathbf{B}$ cannot be calculated via (2), to make the best use of $\mathbf{B}$ we choose its marginal area shown to place the boundary map we use LSB replacement to embed it and original LSBs of marginal area is assembled with messages, i.e., LSB-planes of $\mathbf{A}$, and reversibly embedded into $\mathbf{B}$. In most cases, even with a large embedding rate of the length of boundary map is very short the marginal area of $\mathbf{B}$ is enough to accommodate it.

Numerous parameters such as $LNRNLMRMLPRP$, payloads are embedded into the some estimating errors of black pixels Rb , total embedding rounds$x$, start row SR and end row ER of $\mathbf{A}$ in original image is embedded into marginal area in a similar way these parameters play an important role in data extraction and image recovery process.

*3). Image Encryption:*After rearranged self-embeddedimage, denoted by $\mathbf{X}$, is generated, we can encrypts $\mathbf{X}$ to construct the encrypted image is denoted by $\mathbf{E}$ With a stream cipher the encryption version of $\mathbf{X}$ is easily obtained. For example, a gray value $\mathbf{X}$i,j ranging from 0 to 255, Those 8 bits denoted by $\mathbf{X}$i,j(0),$\mathbf{X}$i,j(1),…..$\mathbf{X}$i,j(7), such that

$$\mathbf{X}_{i,j}(k) = \left\lfloor \frac{\mathbf{X}_{i,j}}{2^k} \right\rfloor \bmod 2, \quad k = 0, 1, \ldots, 7. \qquad (3)$$

The encrypted bits $\mathbf{E}_{i,j}(k)$ calculated through exclusive-or operation

$$\mathbf{E}_{i,j}(k) = \mathbf{X}_{i,j}(k) \oplus r_{i,j}(k), \qquad (4)$$

Where $r_{i,j}(k)$ is generated via a standard stream cipher determined by the encryption key. At last we embed 10 bits information into LSBs of first 10 pixels in encrypted version of $\mathbf{A}$ to tell data hider the number of rows and the number of bit-planes. After image encryption, the data hider or a third party cannot access the content of original image without the encryption key, therefore privacy of the content owner being protected.

B. Data Hiding in Encrypted Image

Once the data hider acquires the encrypted image $\mathbf{E}$ embed some data into it, though he does not get access to the original image. The embedding process is starts with locating the encrypted version of $\mathbf{A}$, denoted by $\mathbf{A}_E$. Since $\mathbf{A}_E$ has been rearranged to the top of $\mathbf{E}$, it is effortless for the data hider to read 10 bits information in LSBs of first 10 encrypted pixels. Afterward significantly how many bit-planes and rows of pixels he can modify, simply the data hider adopts LSB replacement to substitute the available bit-planes with additional data$\mathbf{m}$. Finally, the data hider sets a label following $\mathbf{m}$ to point out the end position of embedding process and further encrypts $\mathbf{m}$ according to the data hiding key to formulate marked encrypted image denoted by $\mathbf{E}_I$. Anyone who does not possess the data hiding key could not extract the additional data.

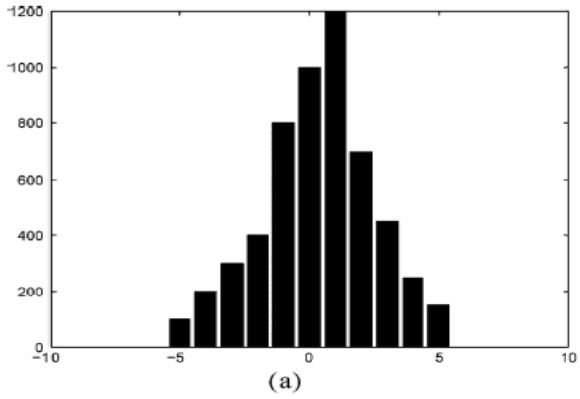

Figure 1: selection of proper points (a). Original Histogram

C. Data Extraction and Image Recovery

The data extraction is completely independent from image decryption, the order of them implies two different practical applications.

Case 1: Extracting Data from Encrypted Images: Tomanage and update personal information of images which are encrypted for protecting clients' privacy, a database manager may only get access to the data hiding key and have to manipulate data in encrypted domain. The data extraction before image decryption guarantees the feasibility of our work in this case. When the database manager gets the data hiding key he can decrypt the LSB-planes of $\mathbf{A}_E$ and extract the additional data $\mathbf{m}$ by directly reading the decrypted version. Later requesting for updating information of encrypted images the database manager will updates information through LSB replacement and encrypts updated information according to the data hiding key all over again. The whole process entirely operated on encrypted domain will avoids the leakage of original content.

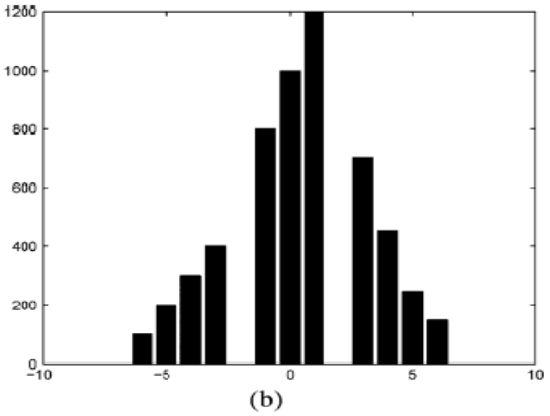

Figure 2: selection of proper points (b). Shifted Histogram

(In this figure, length of messages is 1000 bits, LP = -2 and RP = 2)

Case 2: Extracting Data from Decrypted Images: In the first Case, both embedding and extraction of the data are manipulated in encrypted domain. Further, there is a different situation that the user wants to decrypt the image first and extracts the data from the decrypted image when it is needed. We see the following example is an application for such scenario. Assume that Alice was out-sourced her images to a cloud server, and those images are encrypted to protect their contents. That encrypted images the cloud server marks the images by embedding some notation

includes the identity of the images' owner the identity of the cloud server and time stamps, to manage the encrypted images. The cloud server doesn't have any right to do any permanent damage to the images. Therefore authorized user Bob who has been shared the encryption key and the data hiding key, down-loaded and decrypted the images. Bob is hoped to get the marked decrypted images those decrypted images still including the notation, which that it can be used to trace the source and history of the data. The direction of image decryption before/without data extraction is perfectly suitable for this case. Afterward, we describe how to generate a marked decrypted image..

a) Generate Marked Decrypted Image: To form themarked decrypted image $\mathbf{X}''$ which is made up of $\mathbf{A}''$ and $\mathbf{B}''$ the content owner should do following two steps.

Step 1. With the help of encryption key, the content owner will decrypts the image except the LSB-planes of $\mathbf{A}_E$. The decrypted version of $\mathbf{E}'$ containing the embedded data can be calculated by

$$\mathbf{X}''_{i,j}(k) = \mathbf{E}'_{i,j}(k) \oplus r_{i,j}(k) \quad (5)$$

and

$$\mathbf{X}''_{i,j} = \sum_{k=0}^{7} \mathbf{X}''_{i,j}(k) \times 2^k, \quad (6)$$

Step 2. Extract SR and ER in marginal area of $\mathbf{B}''$. By rearranging A" and B" to its original state that the plain image containing embedded data is obtained.

As can be seen, the marked decrypted image $\mathbf{X}''$ is identical to rearranged $\mathbf{X}$ except LSB-planes of$\mathbf{A}$. At the meantime, it keeps perceptual transparency compared with original image$\mathbf{C}$. More specifically, the distortion is introduced via two separate ways: the embedding process by modifying the LSB-planes of $\mathbf{A}$ and self-reversible embedding process by embedding LSB-planes of $\mathbf{A}$into$\mathbf{B}$. The first part distortion is well controlled via exploiting the LSB -planes of $\mathbf{A}$ only and the second part can benefit from excellent performance of current RDH techniques.

b) Data Extraction and Image Restoration: After generating the marked decrypted image the content owner further extract the data and recover original image. This process is similar to that of traditional RDH methods [10], [11]. The following outlines the specific steps:

Step 1. Record and decrypt the LSB-planes of $\mathbf{AE}$ according to the data hiding key; extract the data until the end label is reached.

Step 2. Extract $LNRNLMRMLPRPR_{bx}$ and boundary map from the LSB of marginal area of $\mathbf{B}''$. Then, scan $\mathbf{B}''$ to undertake the following steps.

Step 3. If R_b is equal to 0, which means no black pixels participate in embedding process, then go to Step 5.

Step 4.calculate the errors *e'i,j* and estimates black pixels *B"i,j* which belongs to [1, 254] to recover the estimating Error and original pixel value in a reverse order and extract embedded bits when *e'i,j* is equal to LN,LM (or LP),RM(or RP) and RN. Else, if *B"i,j* € [1, 254], refer to the corresponding bit *b* in boundary map. If here *b=0*, skip this one or else operate like *B"i,j* € [1, 254], . Repeat this step until the part of payload *Rb* is extracted. If payload *Rb* is extracted bits are LSBs of pixels in marginal area restores them immediately.

Step 5. Calculate estimating errors *e'i,j* of white pixels *B"i,j* and extract embedded bits and recover white pixels in the same manner with Step 4. If these extracted bits are LSBs of pixels in marginal area restores them immediately.

Step 6. Continue doing Step 2 to Step 5 x-1 rounds on *B"*and merge all extracted bits to form LSB-planes of A. Until now, we have perfectly recover B.

Step 7. Replace marked LSB-planes of A" with its original bits extracted from B″ to get original cover image C. We note that if the content owner wants to retrieve his image in Case 1, will do the procedures are exactly the same in Case 2. So, it is omitted in Case 1 for simplicity.

4. Conclusion

Reversible data hiding in encrypted images is a new topic drawing attention because of the privacy-preserving requirements from cloud data management. Earlier methods are implement RDH in encrypted images by vacating room after encryption was opposed to which we proposed by reserving room before encryption. With that the data hider can benefit from the extra space emptied out in previous stage to make data hiding process effort-less. The proposed method can take advantage of all traditional RDH techniques for plain images and achieve excellent performance without loss of perfect secrecy. Moreover, this novel method can achieve real reversibility, data extraction and greatly improvement on the quality of marked decrypted images.

References

[1] T. Kalker and F. M. Willems, "Capacity bounds and code constructions for reversible data-hiding," in *Proc. 14th Int. Conf. Digital Signal Pro-cessing (DSP2002)*, 2002, pp. 71–76.

[2] W. Zhang, B. Chen, and N. Yu, "Capacity-approaching codes for reversible data hiding," in *Proc 13th Information Hiding (IH'2011),LNCS 6958*, 2011, pp. 255–269, Springer-Verlag.

[3] W. Zhang, B. Chen, and N. Yu, "Improving various reversible data hiding schemes via optimal codes for binary covers," *IEEE Trans.Image Process.*, vol. 21, no. 6, pp. 2991–3003, Jun. 2012.

[4] J. Fridrich and M. Goljan, "Lossless data embedding for all image for-mats," in *Proc. SPIE Proc. Photonics West, Electronic Imaging, Secu-rity and Watermarking of Multimedia Contents*, San Jose, CA, USA,Jan. 2002, vol. 4675, pp. 572–583.

[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896,Aug. 2003.

[6] Z. Ni, Y. Shi, N. Ansari, and S. Wei, "Reversible data hiding," *IEEETrans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar.2006.

[7] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.

[8] X. L. Li, B. Yang, and T. Y. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3524–3533, Dec.2011.

[9] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," *Signal Process.*, vol. 89, pp. 1129–1143, 2009.

[10] L. Luo *et al.*, "Reversible image watermarking using interpolation tech-nique," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 1, pp. 187–193, Mar. 2010.
[11] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y.-Q. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Trans.Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 989–999, Jul. 2009.
[12] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook ofApplied Cryptography*. Boca Raton, FL, USA: CRC, 1996.
[13] K. Hwang and D. Li, "Trusted cloud computing with secure resources and data coloring," *IEEE Internet Comput.*, vol. 14, no. 5, pp. 14–22, Sep./Oct. 2010.
[14] M. Johnson, P. Ishwar, V. M. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Trans. SignalProcess.*, vol. 52, no. 10, pp. 2992–3006, Oct. 2004.
[15] W. Liu, W. Zeng, L. Dong, and Q. Yao, "Efficient compression of en-crypted grayscale images," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 1097–1102, Apr. 2010.
[16] X. Zhang, "Reversible data hiding in encrypted images," *IEEE SignalProcess. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.
[17] W. Hong, T. Chen, and H. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Process. Lett.*, vol. 19, no. 4, pp. 199–202, Apr. 2012.
[18] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEETrans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.
[19] Miscelaneous Gray Level Images [Online]. Available:http://decsai.ugr.es/cvg/dbimagenes/g512.php

Author Profile

Chaple. Gopal received the B.Tech degree in computer science and Engineering from JNTU Hyderabad in 2011 and pursuing M.Tech Degree in Computer Science and Engineering from CVSR College of Engineering from ANURAG Group of Institutions, JNTU Hyderabad, India

G. Balram working as assistant professor in Computer Science Engineering from CVSR College of Engineering from ANURAG Group of Institutions, Venkatapur(V), Ghatkesar (M), RangaReddy District, Hyderabad-500088, Telangana State.