Journal of Information Security and Applications 48 (2019) 102368

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

A prediction error based histogram association and mapping technique for data embedment

A.H.M. Kamal[a,*], Mohammad Mahfuzul Islam[b,1]

[a] Department of Computer Science and Engineering (CSE), Jatiya Kabi Kazi Nazrul Islam University, Trishal, Mymensingh, P.C. 2220, Bangladesh
[b] Mohammad Mahfuzul Islam works with Department of CSE, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh

ARTICLE INFO

Keywords:
Reversible
Embedment
Embedding capacity
Stego quality
Prediction errors
Histogram

ABSTRACT

A histogram association and mapping based embedding policy implants a group of bits, called message chunk, in a block. The strategy first calculates a distribution of pixel values of each block in the grayscale, called histogram. Each gray value acts as a bin of the histogram. Though the histogram makes up of 256 bins, only several bins hold these pixel values of a block. The implantation rules shift the pixel value containing bins as a single object to a new position in the histogram. The number of implanted bits is dependent on the range of pixel values of the working block. Implanting bits' number increases for a smaller value of range and vice versa. Being motivated, this research does the identical embedding task by the range of absolute-valued prediction errors. The research implements a predictor and measures the prediction errors in their absolute values. Computed errors' range is smaller than the range of the pixel values. The proposed method does the histogram-shifting task by the range of error values. The proposed method increases the number of embedded bits by a multiple of 1.31 to 2.04. In the second phase, it further applies the same predictor to the last calculated absolute-valued errors. The method yields a smaller range of absolute-valued errors by repeating the procedure for a number of times. The algorithm applies that smaller range in implementing HAM policy. The approach increases the number of implanted bits.

1. Introduction

Image steganography is a technique of implanting secrets in an image, named cover image. An encoder implants the secrets in the cover image by a set of embedding rules. After implanting data, the encoder end sends the adjusted image, known as stego image, on a target end. A decoder at the destination end extracts the secret message from the stego image. Some of the schemes, additionally, rebuild the cover image from the stego image. Based on the capability of rebuilding the cover image from the stego image, the steganography schemes are divided into two types - irreversible and reversible. Irreversible schemes cannot reconstruct the cover image but the secrets only [1–6]. Reversible schemes both extract the secrets and rebuild the cover image from the stego image [7–15]. The reversible schemes carry special importance when the destination end uses the cover image for their further processing task. Moreover, these schemes use more features for maintaining their reversibility. These added features improve implanted data's security [10,11,14]. Reversible algorithms fall into two categories. Methods of the first type try to maximize stego quality. These schemes try to reduce the image distortion level in the stego image. They try to keep the stego excellence close to the cover image [7,12–13,15]. The methods of other category destroy the image quality intentionally [16–20]. Cover information is not accessible in that distorted image. Such methods are useful when the cover contents are also secret [2,25].

The intentionally image distorting schemes destroy the information in the cover image while implanting data. No illicit person or device can rescue the original cover contents from the stego image. Such schemes play important roles in medical and forensic applications; in transmitting legal documents, evidence or report by law-enforcing agencies; in sending copyright and certificates; and in similar purposes when the cover image itself regards as a secret. The schemes achieve the target of distorting the image by one of the following four ways.

i) The multilayer data embedment schemes implant secrets into the contents of multiple levels of the histogram

* Corresponding author.
E-mail addresses: kamal@jkkniu.edu.bd (A.H.M. Kamal), mahfuz@cse.buet.ac.bd, vc@cub.edu.bd (M.M. Islam).
[1] He is currently working as a vice-chancellor of Canadian University of Bangladesh, Dhaka; (Member, IEEE).

https://doi.org/10.1016/j.jisa.2019.102368
2214-2126/

[7,15,21]. While embedding data up to a large embedding layer, the schemes change the pixel values by a big value. Thus, these schemes degrade image quality.

ii) Few methods, like Liao et al. [18] and Zhang et al. [20], first destroy the cover image by encrypting the pixel values of the cover. Then, the schemes embed the secrets in the encrypted values.

iii) Wong et al. use a unified constructive permutation function to shuffle the indices of pixels and then introduce a noise function on the way of distorting an image [26].

iv) A famous scalable degradation method is the histogram association and mapping (HAM) technique [19]. The HAM method first distributes the pixels of an image-block in 0 to 255 marked bins. The distribution forms a histogram. As the block size is small, the pixel value containing bins cover a small region in the histogram. The HAM scheme shifts all non-empty bins as a single object by a value in the histogram. Thus, the scheme changes the pixel values of each block by an equal amount. The range of pixel values fixes the shifting amount of a block.

The first method shifts the pixel values unevenly, i.e., by unequal amounts. The embedding rules change many pixel values by a small number and allow many others to continue as unchanged [7,15,21]. For this, the cover information is partially accessible or fully guessable. The second and third ones are dependent on additional processing tasks for distorting the image, like applying encryption, shuffling indices of pixels and adding noise. Though the encryption process, mentioned in the second case, fully razes the cover information, it works independently. The data hiding process is not methodologically dependent on the encryption technique. It seems that encryption is an additional task of these schemes, however, it increases the processing complexity. In the third one, the image distorting attempts are not part of data embedment. These attempts increase the additional processing costs as well. On the other hand, the fourth case stated HAM based policy distorts the image by its embedding rules. The scheme applies HAM policy to shifts the pixel value histogram of a block as a single object by an amount within the gray color range [9,19]. The scheme confirms an adjustment of all pixel values by an amount. The HAM policy, first, calculates a histogram of the pixel values for each working block. The histogram is a representation of the distribution of pixel values in 0–255. Though, the bins of histogram range from 0 to 255, only a few bins hold the block pixels. It computes the range of non-empty bins. Based on that range, it divides the grayscale into equal sized parts. One gray part holds the histogram bins, known as the origin. The policy next calculates all probable partitions in the grayscale to where the bins of origin could be moved. It uses the value range of the block pixels for computing these probable partitions. The range value also aids in computing the size of embeddable message chunk. Depending on the value of a message chunk, it moves the nonempty bins to the calculated partitions in the histogram as a single object. Thus, the scheme ensures a shifting of histogram bins for each block by an equal value. The process confirms an equal modification of pixel values of a block. Besides, the policy is not dependent on other methods like the encryption process to distort the cover image. It performs that distortions by the embedding rules.

Ong et al. [19] presented the HAM scheme, for the first time, in image steganography. This research chooses the presented embedding policy as a benchmark for four specific reasons. First, the HAM based embedding method is a unique policy of its kind. Second, the method presents a high embedding capacity. Third, it creates enormous distortions in the image. Fourth, the embedding policy is simple. The number of implanted bits by the Ong's scheme, in an image block, is dependent on the range of pixel values of the block. The lower the range is higher the embedding capacity. Nevertheless, the reviewed scheme ignores the matter of reducing the value of the applied range.

This article concentrates on governing the HAM policy by smaller range values. The proposed research presents a new prediction error based HAM scheme. The proposed scheme first applies a predictor to predict all pixel values of a block and then measures the prediction errors in their absolute values. It uses the range of these errors, rather than the block pixels' range, to measure the shifting worth of pixel values of a block. It also employs the range value to calculate the size of embeddable message chunk. As the error's range is smaller than the range of pixel values, the proposed scheme allows more bits for embedding in a block. The experimental results demonstrate an improvement over the existing HAM scheme. The authors publish a part of their contributions in the proceedings of [9], however, the current article presents much improvement over the one in [9]. This article performs HAM policy based on only the min value of the working block, whereas the scheme in [9] does the mapping operation for all the histogram bins as a single object. While [9] was considering the whole histogram as a single object, it was facing problems in handling such blocks which overlap two partitions of the gray parts. That problem is minimized in the current article in a different way. When the stego pixels exceed the gray range, the exceeded pixels are flipped in the schemes. The flipping equation is modified to increase the robustness of the scheme. If the image is divided into n blocks, the scheme of [9] generates 9n bits of side-information whereas the current article produces 8n bits of information for the same. The scheme of [9] takes special care when the range of block pixels is greater than 64, while the current article works in all the blocks in an equal manner. In addition, the current article adds a second proposal which is completely new.

In the second phase, i.e., the second proposal, of the research, the scheme further applies the same predictor on the last calculated absolute-valued prediction errors. It measures the new prediction errors in their absolute values. The scheme repeats the prediction error forming strategy from the last measured absolute-valued errors. After several repetitions, the absolute-valued error values become smaller. Thus, the range of error values becomes smaller. The scheme applies the technique to all the blocks and measures the range values. These smaller range values help the proposed scheme in yielding a notable improvement in implanting more number of bits.

The article presents six sections in the following. Section II briefly describes the scheme of Ong et al. [19] as a benchmark of HAM based embedding method. Section III details the proposed prediction error based HAM (PEBHAM) scheme. Section IV demonstrates and discusses the experimental results. Section V and VI present the proposed repeating PEBHAM (RPBHAM) scheme and the experimental results, respectively. Finally, Section VII concludes the article.

2. The HAM based benchmark scheme

Ong et al. [19] proposed a HAM based data embedment scheme for the first time in the literature. The scheme intentionally destroys the image contents by implanting message bits. The scheme first splits the image of size $h \times w$ into non-overlapping blocks, where h and w are the height and width of the image. Let $m \times n$ is the size of image blocks, where the height and width of the image are divisible by m and n, respectively. While working on a block, the scheme first finds the minimum and the maximum valued pixels in it. Say, their values are the max and min, respectively. The scheme stores these two values as a part of the assistant information, also known as side information. Side information

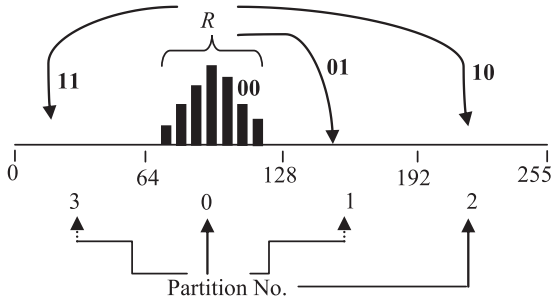

Fig. 1. Grayscale partition and translation of histogram by HAM process. Here, *R* stands for a range value, each bold-marked binary value indicates a probable message chunk.

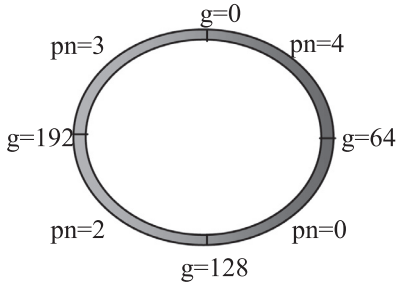

Fig. 2. Grayscale partitioning method, where *g* stands for a gray value at the marked position of the scale and the *pn* represents the partition number.

assists the receiver in the data extraction phase. It measures the range of pixel values, say R, by the relation $R = \max - \min + 1$. The scheme also divides the grayscale range, i.e., 0–255, into P parts, where $P{=}256/2^{\lceil \log_2 R \rceil}$. For example, when $P{=}4$, the gray partitions are (0–63), (64–127), (128–191) and (192–255). The value of P is always 2^i, for $0 \leq i < 8$. It depends on the range value of the block pixels. The scheme calculates a histogram for the block pixels. A partition of the grayscale, known as 'origin partition', holds the pixel value containing bins of the histogram, e.g., (64–127) as shown in Fig. 1. While implanting a message chunk in the block, the data embedment rules of HAM scheme regard the bins in the original partition as a single object. The rules translate that object, i.e., bins of the original partition, to one of the P partitions, known as 'reflective partitions'. The embedding rules mark the original partition as partition no 0. Considering the grayscale as a circular path, as shown in Fig. 2, the scheme visits the partitions by marching in the right direction. While visiting, the scheme marks the other partitions in an incremental way. The scheme separates n bits of the message chunk, say msg, from the message stream, where $n = \log_2 P$. It implants that chunk in the working block. It performs the embedding task in several steps. The scheme converts the bits in msg into a decimal value by a function Bin2Dec(msg). The scheme builds an association of the original partition with the partition no Bin2Dec(msg). The partition no Bin2Dec(msg) is the target partition for translating bins of the original partition. The scheme translates the bins as an object to the target partition. Thus, the interspaces between the translated bins remain unchanged, like the original partition. To understand the method, let the scheme wants to implant a message chunk of '10' in that block. The decimal value of '10' is 2. The implantation rules then shift the bins of partition 0 to partition 2, as shown in Fig. 3. The scheme calculates the stego pixels against the new positions of the translated bins in the histogram.

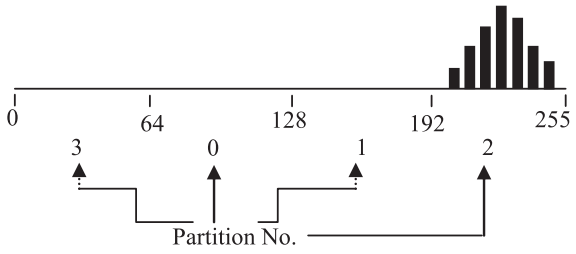

Fig. 3. Histogram shifting of Fig. 1 for implanting a message chunk of 10.

The receiver decodes the stego image to extract the secret message. As it is a reversible scheme, the method rebuilds the cover image as well. For these, the receiver first collects the assistant information from the sender side through another communication channel. The decoder in the receiver end separates the min and max values of the cover block from the assistant information. As the HAM scheme has translated the bins as a single object, the range R_c in the cover block and R_s in stego block are equal. Like the encoder, the decoder determines the values of R (as $R_c{=}R_s$, we term it as R), P and n for the working block successively. It divides the grayscale into P parts and identifies the original and destination partitions. It finds these two partitions by verifying the min values of the cover ($\min_c$) and the stego ($\min_s$). Like the embedding phase, the receiver converts the stego block to the cover block. The decoder forms the cover block by translating the histogram bins of the stego block from $\min_s$ containing partition to $\min_c$ containing partition. It converts the $\min_s$ containing partition number into a binary value. That binary value is the extracted message chunk for the working block.

The HAM scheme presents both a broad-distorted image and a high embedding payload. Embedding payloads talk about the number of implanted bits in an image. Nevertheless, the scheme carries a major drawback. The bins of a block histogram may occupy the area of two partitions. To realize the problem, consider a block whose minimum and maximum values are 80 and 142. The range value of the block is 63 and thus, the total number of partitions is 4. In this scenario, the pixel histogram occupies the region comprising of partitions 0 and 1. Then, the task of defining the original partition is not possible. The authors solve that limitation along with few others in [16]. Still, the literature does not present any attempt to reduce the range of block though it is essential to enhance the embedding payload of the HAM scheme. Section III includes a prediction error based strategy to decrease the value of range and to apply the HAM policy accordingly.

3. Proposed prediction error based HAM scheme

The proposed prediction error based HAM (PEBHAM) scheme applies a predictor to estimate the pixel values of kth block, Bk. The scheme consecutively measures all prediction errors, their magnitude and range of these magnitudes. Let $E_{i,j}$, $E^A_{i,j}$ and R_e denote the prediction errors, error's magnitude and magnitude's range value, respectively. An encoder applies the R_e instead of R to implant a message chunk in the kth block. The R stands for a range of pixel values of a block. The traditional HAM scheme uses the R to implant bits. During data implantation, the proposed scheme translates the pixel values of the kth block by an amount. The R_e acts as a parameter to estimate the translation amount. The scheme produces the stego block from the translated values. After receiving the stego image, the decoder at the receiver end extracts a message chunk from each kth block. At the same time, the decoder rebuilds every cover block from the respective stego block.

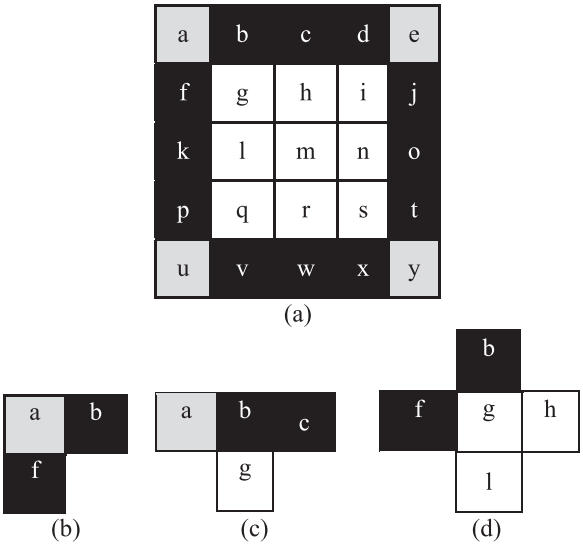

Fig. 4. Prediction process of cover block: (a) a cover block (b) pixel value *a* is predicted by *b* and *f*; (c) *b* is predicted by *a*, *c* and *g*; and (d) *g* is predicted by *b*, *f*, *h* and *l*.

3.1. Predicting block pixels

The proposed PEBHAM scheme chooses a predictor that could create the same prediction errors in both the cover and the stego blocks. A mean value predictor does that wanted task. A mean value predictor estimates a value for a pixel by the statistical mean of associated pixel values in the equation. Though there are many mean value predictors in the literature, for the robustness, the proposed scheme presents a new prediction method. The method estimates the corner pixels, the edge pixels and the inner pixels of a block by the mean of two, three and four neighbor pixels, respectively. Fig. 4(a) shows the inner pixels, the corner pixels and the edge pixels by separating them with white, gray and black color, respectively. Fig. 4(b)–(d) show a snapshot of associated neighbor pixels while estimating the corner pixel a, the edge pixel b and the inner pixel g, respectively. Then, Eq. (1.1), Eq. (1.2) and Eq. (1.3) of Eq. (1) predict the values of a, b and g, respectively.

$$\left.\begin{aligned} P_a &= (b+f)/2 && (1.1)\\ P_b &= (a+c+g)/3 && (1.2)\\ P_g &= (b+h+l+f)/4 && (1.3) \end{aligned}\right\} \quad (1)$$

Following that principality, the scheme measures the prediction values for all the pixels in a block. The scheme computes the prediction errors $E_{i,j}$ by subtracting the predicted values from the corresponding pixel values. Next, Eq. (2) calculates the absolute values of the prediction errors.

$$E^A_{i.j} = ABS\left(E_{i.j}\right) \quad (2)$$

Where the function ABS(.) returns the magnitude of E_i, i.e., ignoring the sign value of E_i.

3.2. Computing error range

The proposed scheme uses the relation $R_e = \max Val(E^A_{i.j}) - \min Val(E^A_{i.j}) + 1$ to measure the value range of errors $E^A_{i.j}$. In that expression, the maxVal and minVal functions return the maximum and minimum of the errors $E^A_{i.j}$, respectively. That range value R_e is smaller than *R*. Like the HAM scheme, the proposed scheme measures a real partitioning value *Rp* and the number of partitions *P*

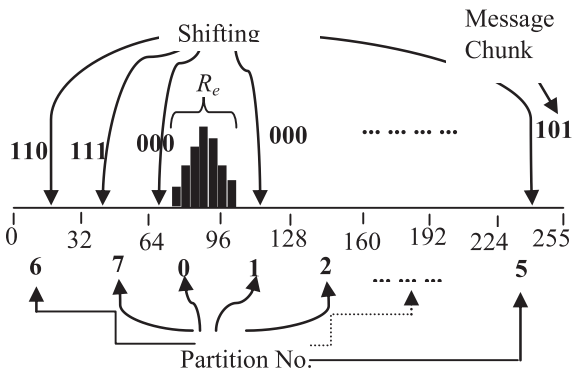

Fig. 5. Prediction error based HAM scheme.

by using Eqs. (3) and (4), respectively.

$$R_P = 2^{\lceil \log_2 R_e \rceil} \quad (3)$$

$$P = 256/R_p \quad (4)$$

The method divides the grayscale into *P* parts and computes a histogram of the block pixels. The scheme embeds $b = 8 - \lceil \log_2 R_e \rceil$ bits of the message chunk in a block. In other words, *b* is $log_2 P$, i.e.,$b = log_2 P$. As R_e is smaller than *R*, the length of b increases for R_e.

3.3. Data embedment process

Let Min_c stands for the minimum value of the pixels in a block. Hence, Min_c is the first bin of the histogram of block pixels. Min_c belongs to a partition in the grayscale. The scheme marks Min_c containing partition by 0. The partition marker marks the next partition by 1 and then the next by 2.

Thus, the partition marker advances by labeling partitions until it tags the rightmost partition in the grayscale. Let, the label of the rightmost partition is *l*. The labeling pointer moves to the leftmost partition in the grayscale and marks the partition by $l+1$. The partition marker moves the pointer to every subsequent partition and provides a partition number. At each movement, the method increases the partition number by 1 until it reaches to the partition number 0. Thus, starting from $l+1$, it labels the next partitions by $l+2$, $l+3$, ..., $P-1$. The range of prediction errors $R_e$helps in determining the number of partitions, to be made in the grayscale. According to the partition numbering rules, the scheme labels the partitions by 6, 7, 0, 1, 2, 3, 4 and 5 from left to right, in Fig. 5. In that figure, Min_c=73 and P=8. Hence, for this block, the size of the embeddable message chunk is 3 because $b = \log_2 P$.

Fig. 5 depicts the gray partitions for a sample value of R_P=32 and histogram of block pixels.

3.4. Data embedment process

Let Min_c stands for the minimum value of the pixels in a block. Hence, Min_c is the first bin of the histogram of block pixels. Min_c belongs to a partition in the grayscale. The scheme marks Min_c containing partition by 0. The partition marker marks the next partition by 1 and then the next by 2. Thus, the partition marker advances by labeling partitions until it tags the rightmost partition in the grayscale. Let, the label of the rightmost partition is *l*. The labeling pointer moves to the leftmost partition in the grayscale and marks the partition by $l+1$. The partition marker moves the pointer to every subsequent partition and provides a partition number. At each movement, the method increases the

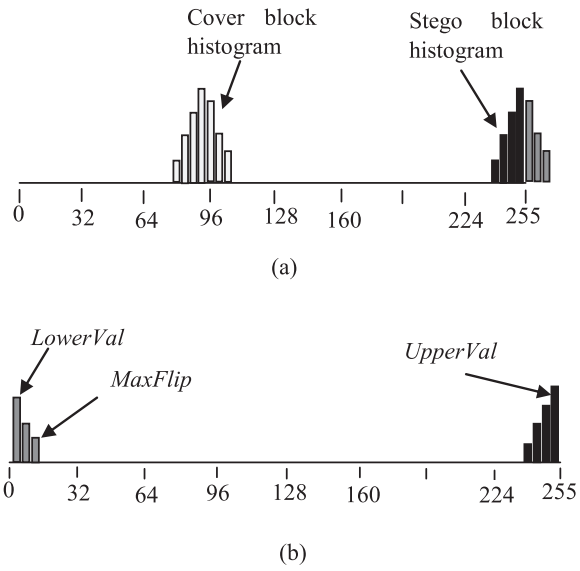

Fig. 6. Stego block generation: (a) Stego block histogram after applying the Eq. (5); and (b) Stego block histogram after applying the Eq. (6).

partition number by 1 until it reaches to the partition number 0. Thus, starting from $l+1$, it labels the next partitions by $l+2$, $l+3$, ..., $P-1$. The range of prediction errors $R_e$helps in determining the number of partitions, to be made in the grayscale. According to the partition numbering rules, the scheme labels the partitions by 6, 7, 0, 1, 2, 3, 4 and 5 from left to right, in Fig. 5. In that figure, Min_c=73 and $P=8$. Hence, for this block, the size of the embeddable message chunk is 3 because $b=log_2P$.

Let, generally, define m as an embeddable message chunk of b-bits and d as its converted decimal value. The scheme makes an association of the partition number 0 with the partition number d and maps the value Min_c to its associated value Min_s in the partition d. It shifts the Min_c valued bin to the mapped Min_s bin in the dth partition. It shifts other bins of the histogram of the block pixels accordingly, i.e., Min_c+1 valued bin to Min_s+1 valued bin, Min_c+2 bin to Min_s+2 bin and so on. The scheme forms the stego block S^kusing Eq. (5), where T_0 and T_d are the starting value of partition no. 0 and d, respectively. Several pixels in S^k may exceed the extreme gray value 255 if max $+T_d-T_0 > 255$, as illustrated in Fig. 6(a). That is why, S^k needs to be processed further. For this, S^k is called unprocessed stego block. The scheme forms the final stego block $\tilde{S}^k$ by flipping these extreme values using Eq. (6). Fig. 6(b) depicts the stego block histogram.

$$S^k = B^k + T_d - T_0 \tag{5}$$

$$\tilde{S}^k = \mathrm{mod}\left(S^k, 256\right) \tag{6}$$

where the function mod(.) returns the remainder value when the second argument divides the first argument.

3.5. Defining parameters for realizing flipped blocks

If the block experiences a flipping procedure, the Eq. (6) sends the stego pixels to two extremes of the grayscale, i.e., near to 0 and 255, as shown in Fig. 6(b). For the purpose of data extraction and cover image reconstruction in a block, the extractor should be able to realize whether the block had experienced a flipping operation or not. If the scheme realizes it as a flipped block, the decoder should also be able to identify the flipped pixels. For this, it measures three extra values LowerVal, UpperVal and MaxFlip by the following four steps.

Step 1: Set $k=1$, n=total number of blocks in the image, *MaxFlip*=0, *LowerVal*=0, *UpperVal*=255.
Step 2: *Set Big*=max(S^k), where max is a function that returns the maximum of its input arguments.
Step 3: If *Big*>255 then [That is, a flipping is demanded.]
$A = \mathrm{mod}(Big, 256)$
$B = \min(\tilde{S}^k)$ [min function returns the minimum]
$C = \max(\tilde{S}^k)$
If A>*MaxFlip* then
MaxFlip=A
End if
If B>*LowerVal then*
LowerVal=B
End if
If C<*UpperVal* then
UpperVal=C
End if
End if
Step 4: If $k<n$ go to step 2

The *LowerVal, UpperVal* and *MaxFlip* act as a part of side-information. Each one is a single value feature for the entire image. Therefore, the method increases the size of side-information by 8 bits for everyone and 24 bits in total, which is a negligible figure. The first two parameters help the decoder to decide whether a block undergoes a flipped procedure or not. If it detects a flipping action in a block, it uses the last parameter to identify the flipped pixels. Finally, the scheme forms the stego image $\tilde{I}$ by concatenating all the stego blocks $\tilde{S}^k$. Example 1 describes the proposed embedding method.

3.6. Side information

The competing scheme [19] requires the minimum and the maximum value of each block as a side-information. The scheme also stores two more information, when $R > 127$. Thus, it adds 4×8-bits, (i.e., 8 bits for each of the minimum, maximum and two others), to the side-information for each block. On the other hand, the proposed scheme stores the value of Min_c as a side—information for each block, i.e., 8 bits of information. In addition, scheme stores 24-bits of information for LowerVal, UpperVal and MaxFlip in its final step for single time only. If the scheme divides an image into n blocks, the size of side-information becomes $4n\times8$ bits in [19] and $n\times8+24$ bits in the proposed scheme. Thus, the proposed scheme produces the side-information of one-fourth of [19]. The decoder end needs that side-information for extracting the implanted information. There are many ways to send the side-information to the destination end [14,15,19,21]. We, like the competing scheme, propose to send the side-information to the destination end through another communication channel before starting the data extraction task.

3.7. Data extraction and cover image reconstruction process

The decoder collects the values of Min_c, *LowerVal, UpperVal* and *MaxFlip* from the side-information. The scheme measures the minimum and the maximum values in a stego block $\tilde{S}^k$. Let, these two values are min and max, respectively. The scheme realizes the necessity of performing a flipping operation in a stego block when $min \le LowerVal$ and $max \ge UpperVal$. The scheme next identifies the flipped pixels and returns them to their not-flipped position $S^k_{i,j}$ by Eq. (7). Thus, it reforms $S^k_{i,j}$ from $\tilde{S}^k$.

$$S^k_{i,j} = \begin{cases} \tilde{S}^k_{i,j} + 256 & \text{if } \tilde{S}^k_{i,j} \le MaxFlip \\ \tilde{S}^k_{i,j} & \text{Otherwise} \end{cases} \tag{7}$$

The scheme applies the same predictor to the pixels of $S^k_{i,j}$. After that, it measures the values of $E^k_{i,j}$, R_P and P. The method applies Eq. (8) to measure the value of stego displacement dSk in $\tilde{S}^k$

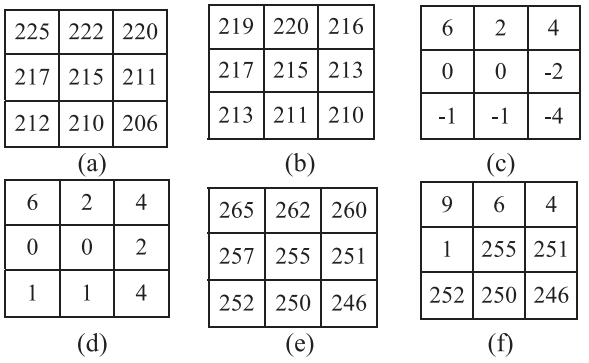

Fig. 7. Data embedment in a block: (a) image block; (b) the predicted values; (c) the prediction error E; (d) absolute error values $E^A_{i,j}$; (e) results of the Eq. (5) $S^k = B^k + T_d - T_0$; and (f) the stego values computed by the Eq. (6).

about the cover block. It uses Eq. (9) to rebuild the cover block $B^k_{i,j}$. Finally, Eq. (10) creates the decimal value d.

$$dS^k = \min\left(S^k_{i,j}\right) - Min_c \tag{8}$$

$$B^k_{i,j} = S^k_{i,j} - dS^k \tag{9}$$

$$d = 2^{\left\lfloor \log_2(dS^k / R_P) \right\rfloor} \tag{10}$$

The scheme converts the d into a binary value. That converted binary is the secret message chunk m. If the size of m is not b-bits, the scheme attaches plenty of 0 s to the left of m. Example 2 explains the whole procedure.

The decoder repeats the procedure in every block to extract the secrets and to reform all cover blocks. The authors publish a part of the work in [9].

3.8. Solving the flipping detection anomalies for the wider block range

If the chunk of the message consists of only 0 s (i.e., message chunk consists of only one or more zeros), the decimal value of the chunk is 0, i.e., $d = 0$. For $d = 0$, the block remains unaltered. Thus, the scheme retains the pixel values of a block as unaltered for implanting a bit value of 0 in it. In that stego block, if $R_e > 128$, the minimum of the stego pixels is close to 0. Similarly, the maximum of these pixels is near to 255. In that scenario, the block pixel may satisfy that $min \leq LowerVal$ and $max \geq UpperVal$. Then, though it is not a flipped block, the receiver detects the block as a flipped one. To solve that problem, the encoder of the proposed method assigns 255 to the side-information parameter Min_c rather than the real min value of the block pixels. If the receiver end finds Min_c=255 and $R_e \geq 128$, the decoder extracts a bit value 0.

Example 1. Implanting a message stream $m = 00101$ into an image block, as shown in Fig. 7(a).

Fig. 7(a) is a sample block where the minimum value is 206, i.e., Min_c=206. The proposed scheme applies a predictor to the block pixels and evaluates the predicted values, the prediction errors and absolute values of the errors successively. Figs. 7(b)–(d) show these values, respectively. Fig. 7(d) provides $Re = 7$. Next, it calculates $Rp = 8$, $P = 32$ and $b = 5$. The decimal value of pointed message chunk m is 5, i.e., $d = 5$. The Min_c=206 is in partition range 199–206. Therefore, partition number 0 is 199–206. The partition number 0 makes an association with partition number 5. The range of partition number 5 is 239–246. Therefore, T0 = 199, Td = 239 and $T_d - T_0 = 40$. According to the Eq. (5), S^k= Bk + 40. Fig. 7(e) shows the not-flipped stego block S^k. Finally, Eq. (6) performs the flipping procedure. Fig. 7(f) shows the final stego block. It stores the values of Min_c, *LowerVal*, *UpperVal* and *MaxFlip* as a side-information. On that block, Min_c=206, *LowerVal*=1, *UpperVal*=255 and *MaxFlip*=9.

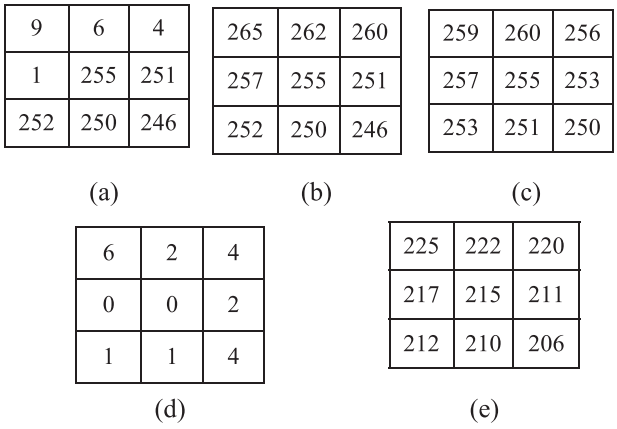

Fig. 8. Message extraction and cover block reconstruction: (a) stego block; (b) flipped stego block yield by applying Eq. (7); (c) predicted stego block; (d) stego prediction errors; (e) cover block constructed by Eq. (9).

Example 2. Message extraction and cover block reconstruction

The decoder end first collects the values of Min_c=206, *LowerVal*=1, *UpperVal*=255 and *MaxFlip*=9 from the side information. Fig. 8(a) depicts the stego pixels of Example 1 created a stego block. The min and max values are 1 and 255 respectively. The block is a flipped one because $min \leq LowerVal$ and $max \geq UpperVal$. The Eq. (7) reforms the not-flipped stego pixels S^k. Fig. 8(b) depicts these S^k values. The scheme applies the same predictor as of the sender side. It measures the predicted values and prediction errors, as shown in Fig. 8(c) and 8(d), respectively. From these prediction errors, it measures the values of $E^A_{(i,j)}$, R_P, P and b. The Eq (8) calculates the stego displacement value dSk. Finally, Eq. (9) rebuilds the cover block, as shown in Fig. 8(e). The Eq. (10) calculates the value of d. The calculated values of b and d are 5 and 5, respectively. The binary equivalent of d is 101. As the length of the message chunk is 5, because of $b = 5$, it places two more zeros at the start of 101. Then, the 5-bits message chunk is 00101. Thus, the scheme extracts the message chunk m

4. Result analysis and discussions

This research verifies the proposed scheme by experimenting on 500 BOSS images [22], 5000 CalTech images [23] and 50 standard images [24]. The scheme first resizes the images to a dimension of 240×240. After the data embedment task, the scheme measures the embedding payload and the value of the peak signal to noise ratio (PSNR). It investigates and analyses these results for various sizes of image blocks. The proposed research follows three major targets - to apply prediction error based HAM policy in the embedding phase, to enhance the embedding payloads and to destroy the image quality. The method implements the prediction error based HAM policy successfully. Looking at the other two issues, the research compares the results of the proposed scheme with its competing schemes [18–19]. To the best of the author's knowledge, Ong et al. [19] first time apply the HAM technique to embed data. The authors of this article presented two amendments [9,16] on the benchmark scheme [19]. Still, there is no other HAM used data embedment scheme in the literature. As a HAM used embedding policy, the research compares its results with the results of the benchmark scheme [19]. The research also incorporates the scheme of Liao et al. [18] and compares its results with the

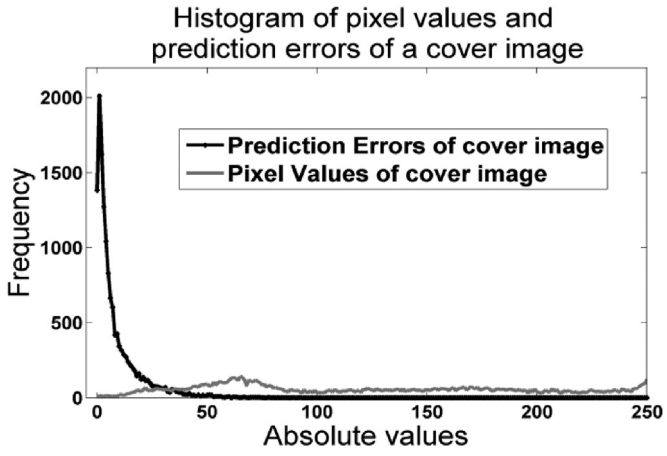

Fig. 9. Histogram of pixel values and prediction errors computed from an image of the CalTech image dataset.

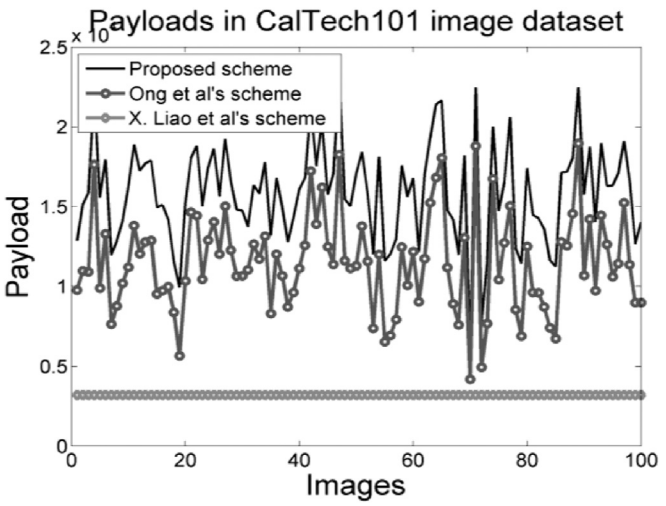

Fig. 10. Comparison of payloads of the proposed scheme with the schemes of Ong et al. and Liao et al. The figure presents results for the first 100 images of the CalTech image dataset.

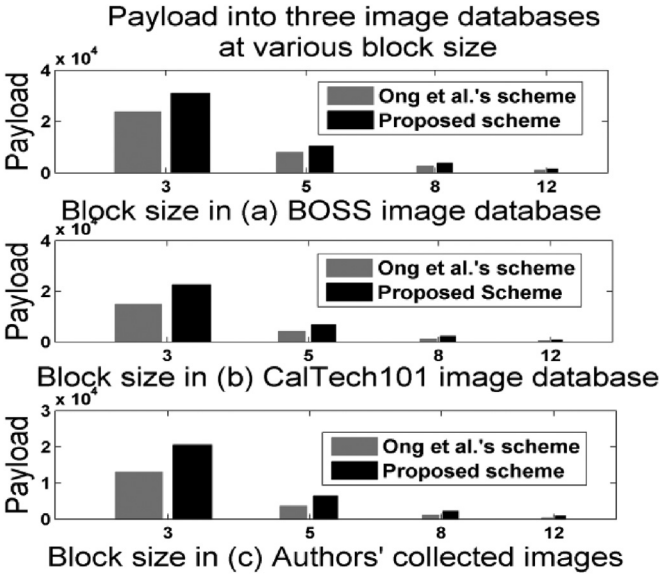

Fig. 11. Average payload obtained into three image databases for different sizes of blocks.

proposed scheme. Though Liao et al.'s scheme does not apply the HAM policy, it destroys the image quality intentionally. Besides, the proposed scheme uses the absolute value of the prediction errors in its embedding procedure, while the scheme of Liao et al. uses the absolute differences of neighbor pixels to control their data embedment task. This section presents the results to justify the claim of boosting up the embedding payloads and improving the image distortions by the proposed scheme.

4.1. Justification of applying the range of errors in the HAM policy

To provide justification for using error's range in HAM scheme, the research presents Fig. 9. The figure puts forward distribution of the absolute-valued prediction errors and the pixel values of a sample image. A major part of the prediction errors takes places in the bins which are close to '0′, whereas the pixel values scatter themselves over the gray range about at a flat rate. The affair tells that the bulk of the errors are of smaller valued. Therefore, it implies that the range, R_P of the absolute-valued prediction errors of a block is smaller than the range, R of their pixel values.

The HAM used schemes, both [19] and the proposed scheme, embed less number of bits in a block when the range value becomes larger. The schemes embed a bit in a block when the range value is greater than 64 because then $P=2$ as $P=256/2^{\lceil \log_2 R \rceil}$. The research measures *Rp* of the absolute-valued prediction errors and R of the pixel values for each block. It conducts the experiment in the images of three different image datasets and for various sizes of image blocks. The research finds all blocks with a range of 64 or greater. It counts and tabulates that information in Table 1 separately for [19] and the proposed scheme. The results reveal that the number of large ranged blocks in [19] is about 3–6.36 time of the proposed scheme.

Table 1
Average number of blocks whose block range is greater than 64.

Image database	Scheme	Block size			
		3 × 3	5 × 5	8 × 8	12 × 12
BOSS	Ong et al.	435	312	189	115
	Proposed	110	69	41	27
CalTech	Ong et al.	1266	776	416	240
	Proposed	388	236	137	80
Standard	Ong et al.	1136	801	455	249
	Proposed	198	126	77	49

4.2. Analysis of embedding payloads

The research assesses the payloads in different image datasets and schemes. Fig. 10 sketches the results for the Caltech image dataset only. The figure depicts the payloads for the first 100 images along its y-axis.

Fig. 10 states that the proposed scheme dominates the other schemes noticeably by the achieved payloads. Among the compared schemes, Liao et al.'s one [18] presents the lowest embedding payloads because the method implants a single bit in each image block. Ong et al.'s scheme [19] presents a bit improved payloads because of their tries of embedding multiple bits in each image block. The proposed method improves Ong et al.'s embedding payloads by a multiple of about two. As only Ong et al.'s scheme competes with the proposed scheme for its payloads, this research compares these two schemes further in Fig. 11 for different sizes of image blocks. The figure indicates the average payloads for three image datasets in three different subfigures. It shows the average payloads along the y-axis. The figure groups the re-

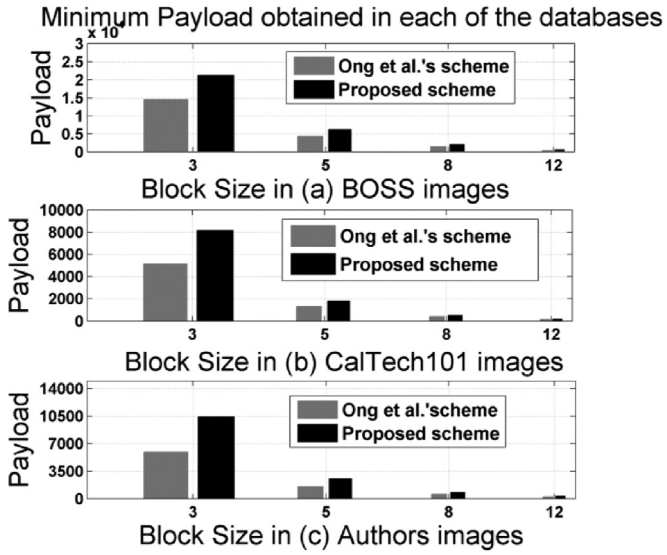

Fig. 12. Minimum payloads obtained in each of the image datasets.

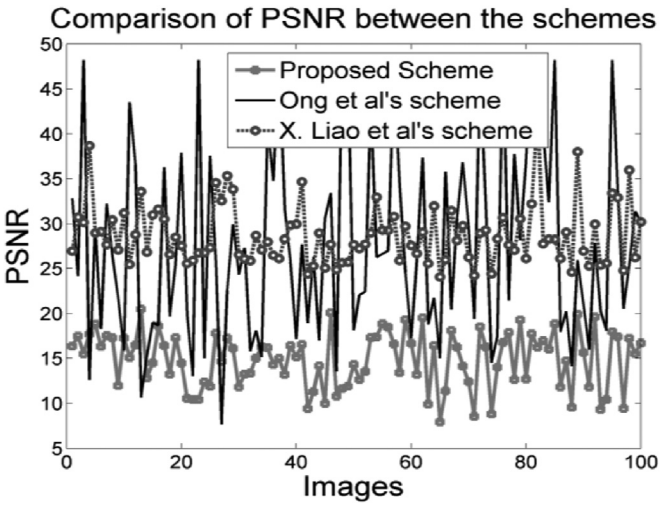

Fig. 13. Comparison of PSNRs achieved in the first 100 images of the CalTech image dataset by different schemes.

sults of the proposed scheme and the Ong et al.'s scheme for each block size along the x-axis. Embedding payloads of the proposed scheme are greater than the obtained one in Ong et al.'s scheme for all sizes of image blocks. Achieved payloads for blocks of size 3×3 is higher than the payloads for other sized blocks because of their block's number. The average payloads increases in the proposed scheme about the Ong et al.'s scheme in the image dataset of BOSS, CalTech and Standard by a multiple of {1.31, 1.51, 1.58} for block size of 3×3, {1.33, 1.62, 1.77} for block size of 5×5, {1.35, 1.73, 1.93} for block size of 8×8 and {1.38, 1.83, 2.04} for block of 12×12. As all the multiples are more than 1, it proves a definite improvement. Improved multiples vary from 1.31 to 2.04 depending on the categories of images and sizes of image blocks.

The research also compares the lowest earned payloads among the schemes. The research calculates the minimum payload for each image dataset and different sizes of image blocks. Fig. 12 presents the results. The figure demonstrates that the proposed scheme dominates the competing schemes by a multiple of 2 or more. The proposed scheme always provides higher embedding payloads. Thus, this research settles its superiority over the others.

4.3. Analysis of PSNR value

The target of the proposed scheme is to destroy the image quality intentionally and notably. To test the distortion level statistically, this research also measures the PSNR values of the resulted stego images.

Figs. 13 and 14 present the values of PSNR. Fig. 13 shows the calculated PSNR values for three different schemes and for the first 100 images of the CalTech image dataset. On the other hand, Fig. 14 presents the PSNR values in three different image datasets for various sizes of image blocks. Liao et al.'s scheme [18] presents the highest values for PSNR in all the images, as shown in Fig. 13, because it does not change the values of 50% contents of the embedding space according to their embedding rules. On the other hand, PSNR values in the Ong et al.'s scheme [19] and in the proposed scheme depend on the pattern of the implanted bits and the computed block ranges. For this, the PSNR values of these two schemes vary from image to images. Again, the flipped procedure changes the values of many pixels from very large values to very small values in the proposed scheme. Thus, the proposed

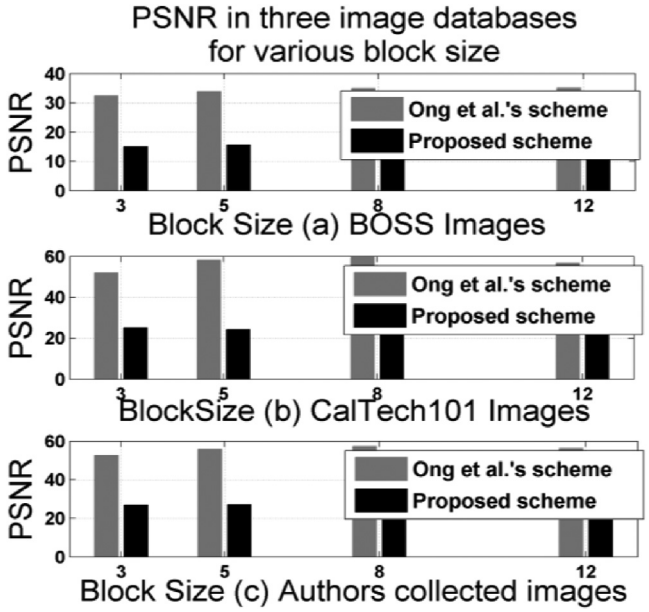

Fig. 14. PSNR for different image datasets at different block sizes.

scheme damages the correlations among the block pixels and does structural distortions where the Ong et al.'s scheme does not do that. As a result, the proposed scheme presents worsening PSNR values.

The PSNR values decrease for smaller sized image blocks, as shown in Fig. 14. In a smaller sized image block, the range value decreases. When the range value is small, the number of gray partitions increases. Increased number of partitions enlarges the chance of associating cover block with more number of partitions in the HAM policy. It, indeed, widens the size of the implanted message chunk. As the size of the message chunk increases, the number of only '0' bit consisted message chunk decreases. If the chunk does not consist of only '0's, that chunk causes a certain change in the stego pixels.

For these stated reasons, the proposed scheme gives smaller PSNR value than the Ong et al.'s scheme

5. Further improvement of embedding capacity by applying repeated prediction process

The results presented in Section IV state that the payload of the proposed scheme increases for a smaller value of R_P. Realizing that, this section presents a repeated prediction policy to reduce the magnitude of the range value R_P for each block in the prediction error space. In Section IV, the PEBHAM scheme produces the $E^A_{i,j}$ values. In this section, repeated PEBHAM (RPBHAM) scheme applies the same predictor for n times to reduce the value of R_P. The prediction process of RPBHAM works for n-times in a block as following 8 steps.

Step-1	Set $t=1$
Step-2	Select the same predictor P that the PEBHAM has applied.
Step-3	Measure the $E^A_{i,j}$ values. The RPBHAM scheme deems the $E^A_{i,j}$ values as the pixel values of an image, i.e., $E^A_{i,j}$ acts a virtual image plane.
Step-4	The RPBHAM applies the predictor P to the virtual image plane $E^A_{i,j}$ to estimate the values of $E^A_{i,j}$.
Step-5	It measures the prediction errors and then their absolute values. Let, $E^R_{i,j}$ is the absolute error values.
Step-6	The RPBHAM scheme sets $E^A_{i,j}$=$E^R_{i,j}$.
Step-7	If $t < n$, the RPBHAM scheme starts again from step 3.
Step-8	It measures the range of $E^A_{i,j}$. Let, the range value is R^R_P

The value of R^R_P is smaller than R_P. The RPBHAM method assigns the value of R^R_P to R_P, i.e., R_P=R^R_P, to apply the same embedment method of PEBHAM. At that stage, the RPBHAM scheme applies the data implantation that the Section III states. Like the PEBHAM method, the RPBHAM scheme adds all Min_c values, LowerVal, UpperVal and MaxFlip values as a side-information. The scheme also assigns n as a part of the side-information. The scheme sends the side information to the destination through another communication channel.

During the data extraction phase, the scheme first finds all the Min_c values and the values of n, LowerVal, UpperVal and MaxFlip from the collected side information. It next applies RPBHAM prediction method for n-times and then measures the value of R_P. After finding the value of R_P, RPBHAM scheme applies the data extraction procedure of Section III.

6. Result analysis of RPBHAM scheme

The research examines the RPBHAM scheme in different image datasets. Figs. 15 and 16 sketches the results of payloads and PSNRs, respectively. Both the Fig. 15(a) and (b) present the payloads of Ong et al.'s scheme and the RPBHAM scheme for different values of n in the image datasets of CalTech and BOSS respectively. The payload increases with the increment of n, as illustrated in Fig. 15(a) and (b). The figures draw the consequences for 100 images. The Fig. 15(a) states that, regarding the competing scheme [19], the proposed scheme achieves 2.8, 3.1, 3.4 and 3.6 times higher payloads in many images for applying the predictor for 3-times, 5-times, 7-times and 9-times, respectively. Fig. 15(c) displays the average payloads for all the images in each image dataset. The figure shows the same fact that the proposed scheme enhances the embedding capacity and it increases for raising the value of n. Fig. 16 analyses the PSNR values. In RPBHAM scheme, PSNR value decreases for increasing the value of n, as shown in Fig. 16(a). Fig. 16(b) shows the average of PSNR values. All the average PSNR values are small for the RPBHAM scheme. It means the RPBHAM scheme deeply destroys the cover information than the competing scheme.

To test the structural similarity between the cover and the stego image, the research also implements the structural similarity index value measuring method of [27]. To know details, the reader is requested to read the article of [27]. The experiment computes the values of SSIM. A smaller value of SSIM indicates a larger structural distortion in the stego image. The average SSIM values for all the image dataset are computed and tabulated in Table 2. The results state that the proposed scheme provides smaller values of SSIM in all the image dataset. Thus, it ensures that the proposed scheme performs more structural distortion in the stego image and that happens for executing the flipping operation by the proposed scheme.

Table 2
Average SSIM values in three image datasets.

Schemes	CalTech101	BOSS	Standard
Ong et al.	0.293	0.285	0.260
RPBHAM	0.230	0.2452	0.226

7. Discussion for REPHAM scheme

7.1. Robustness of the scheme against brute force attacks

Intruders may try to retrieve the secrets as well as to reconstruct the cover image from the stego by unauthorized access to the stego image. A statistical measure of the possible trials that are required to break the security of implanted data and the cover image is computed for the proposed method. A total number of trials are defined by Eq. (11).

$$\prod_{i=1}^{4} H_i \tag{11}$$

where,

H_1: It defines the number of blocks that could be generated in an image for different block sizes. The proposed scheme divides the image of size $M \times N$ into blocks of $m \times n$, where m, $n \geq 3$. That is, for $m=3$, n could be any value between 3 to N. Similarly, for $m=4$, n could be any value between 3 to N. Hence, the total possible number of different image blocks are $(M-3)\times(N-3)$.

H_2: It represents the number of times for what the scheme applies the prediction policy. Say, it is r, where $r > 0$.

H_3: It states the possible number of places in the grayscale to where the Min_c of the block histogram could be mapped. Based on the range value, grayscale is partitioned into 2^k parts, where $k \in [1, 8]$. When, the gray partition is 2, i.e., for $k=1$, the histogram could be moved to two places by the HAM policy. It could be shifted to 4 places for 4 number of gray partitions, i.e., for $k=2$, and so on. An intruder has to check all the possibilities. Thus, different possible gray partitions are $\sum_{k=1}^{8} 2^k$, which is the value of H_3.

H_4: It indicates the highest number of attempts that are required to detect a necessity of flipping operation. The proposed scheme uses LowerVal, UpperVal and MaxFlip values to identify a flipped stego block and to return the flipped values to their original positions. The values of these three variables vary from 1 to 127, i.e., each one can hold value within the range [1, 127]. Thus, the total combination of holding values by these three variables are $127 \times 127 \times 127$, i.e., 127^3.

The discussions reveal that a total number of combinations of trials for a brute force attacks are $(M-3)\times(N-3)\times \mathrm{r} \times \sum_{k=1}^{8} 2^k \times 127^3$. It is a big number. In practice, it impossible for an intruder to retrieve the implanted secrets and to reconstruct the original image from a brute force

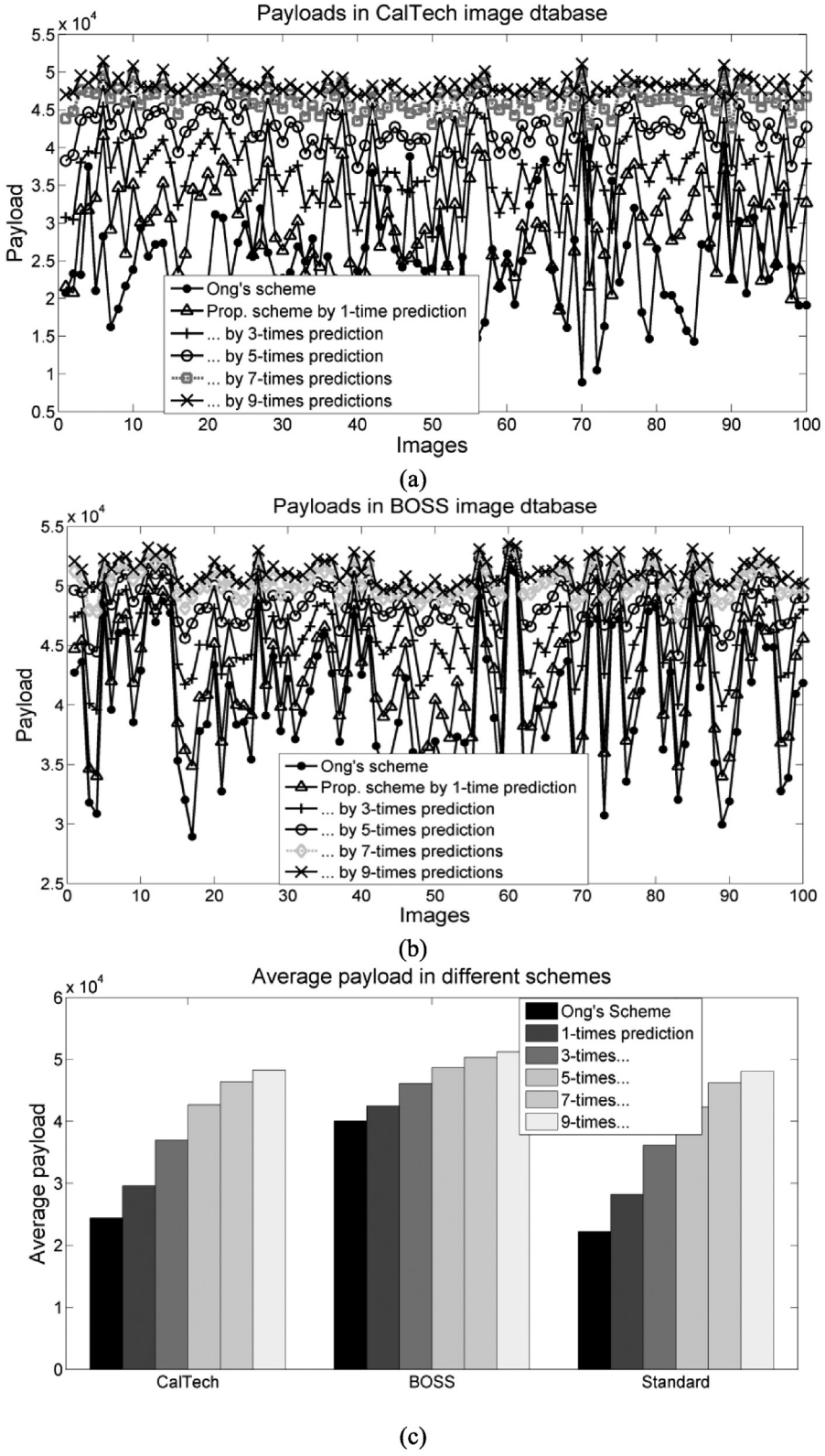

Fig. 15. Analysis on payloads in different image dataset: (a) the Caltech; (b) the BOSS; (c) the Standard image datasets.

attack because, in a worst case, it will require several millions of year to be succeeded in an image of 510×510. The possibility of success by an unauthorized client will increase by $(M-3)\times(N-3)$ if we define LowerVal, UpperVal and MaxFlip for each block separately, instead of the whole image. The number will again increase by $(M-3)\times(N-3)$ if we allow varying the iteration number of applied predictions from block to block. If it is, the combination of trials will be $((M-3)\times(N-3))^3 \times \mathrm{r} \times \sum_{k=1}^{8} 2^k \times 127^3$.

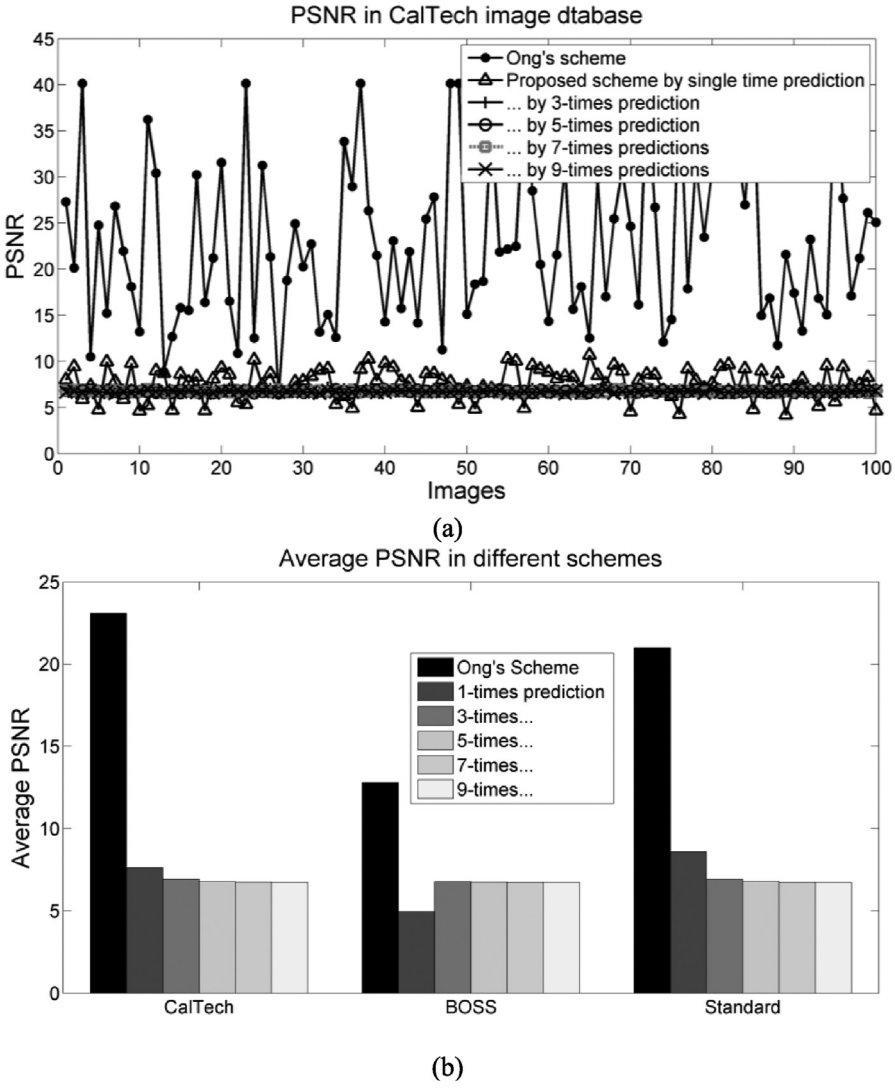

Fig. 16. PSNR in different image dataset: in (a) the CalTech; (b) the Standard image datasets.

7.2. Verifying the scheme's capability of performing reversibility

The proposed scheme implants secrets in each block B^k. After data embedment, the scheme generates a stego block $\tilde{S}^k$. At the receiver end, the decoder extracts the secrets from the stego $\tilde{S}^k$. At the same time, it rebuilds each cover block. Say, the decoder generated cover block is D^k. If the reversibility of the method works properly, the blocks B^k and D^k for each k should be equal. Let, these B^k and D^k present the cover image and the decoder generated cover image, respectively, when we ignore the superscript k. The mean square error (MSE) of these two images is computed by Eq. (12).

$$MSE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(D_{i,j} - B_{i,j}\right)}{M \times N} \tag{12}$$

where M and N state the image dimensions.

If B and D become equal, each of the pixels of B and D will coincide. In that case, the value of MSE will be zero. That MSE is used in the experiment to test the reversibility. We check it for all the images of every image dataset and for various sizes of image blocks. Table 3 articulates that none of the images engenders a non-zero MSE value. Thus, the experiments establish the capability of our scheme in performing reversibility, successfully.

Table 3
Non-zero MSE producing images for different image dataset and size of image blocks.

Image database	Block size			
	3×3	5×5	8×8	12×12
BOSS	0	0	0	0
CalTech	0	0	0	0
Standard	0	0	0	0

8. Summary and comments

Image quality degrading based embedding schemes are advantageous when the carrier is a secret. Many cover image carries its own secret information. For this, these schemes destroy the cover image by either data implantation rules or encrypting it. Nevertheless, encryption is not a part of the data embedment mission. Rather the schemes apply an encryption task at the preprocessing stage just to destroy the cover contents. Therefore, it seems as an additional task. On the other hand, destroying an image by embedding rules reduces that extra processing cost, however, performs the objective. These methods allow their embedding rules to damage the image contents at a large scale. As these schemes

are free from managing the stego image quality, the methods find much scope to embed more data bits. Hence, the usability of such methods is increasing day by day. The prediction error based HAM scheme has improved the embedding capacity notably. Finally, the repeated prediction error based scheme has boosted up the embedding performance further. The results are promising. The two proposed methods enhance the embedding capacity by several multiples of its competing schemes. The image degradation by the proposed schemes is equally noticeable. The presented scheme will arise as a notable scheme in the arena of image steganography for implanting large volume of secret data.

Declaration of Competing Interest

The authors do not have any economic interest from that article. The first author was a PhD student and working under the supervision of the second author. The first author has completed his PhD degree. The authors are now publishing their unpublished works. Therefore, the authors have chosen this journal to publish one of the works. Both authors are aware of that submission. The first author was a fellow of ICT division of the Ministry of Post, Telecommunication and Information Technology of the Government of Bangladesh. However, the fellowship neither covers any publication charge nor claims any financial interest from that research.

CRediT authorship contribution statement

A.H.M. Kamal: Conceptualization, Formal analysis, Methodology, Writing - original draft, Writing - review & editing. **Mohammad Mahfuzul Islam:** Formal analysis, Supervision, Writing - review & editing.

Author's contribution

The first author, AHMK, was a PhD student at the Department of Computer Science and Engineering of the Bangladesh University of Engineering and Technology. He was working under the supervision of the second author, MMI. This work is part of the PhD research. Hence, the entire work is supervised and guided by MMI.

Acknowledgments

The author AHMK was funded by the ICT division of the Ministry of Post, Telecommunication and Information Technology of the Government of Bangladesh through a fellowship program during his PhD study. Therefore, the authors like to acknowledge the stated ministry of Bangladesh.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.jisa.2019.102368.

References

[1] Hong Wien, Chen Tung-Shou. A novel data embedding method using adaptive pixel pair matching. Inf Forens Secur IEEE Trans 2012;7.1:176–84.
[2] Kamal AHM, Mahfuzul Islam M. Facilitating and securing offline e-medicine service through image steganography. Healthc Technol Lett 2014;1.2:74–9.
[3] Kamal AHM, Islam MM. Enhancing the performance of the data embedment process through encoding errors. J Electron 2016;5.4:79–95.
[4] Kamal AHM. Securing the smart card authentications process by embedment random number of data bits into each pixel. Int J u- e- Serv Sci Technol 2017;10.7:43–54.
[5] Lin Ching-Chiuan. An information hiding scheme with minimal image distortion. Comput Stand Interfaces 2011;33.5:477–84.
[6] Yin Zhao-Xia, et al. Second-order steganographic method based on adaptive reference matrix. IET Image Proc 2015;9.4:300–5.
[7] Kamal AHM, Islam MM. Boosting up the data hiding rate multi cycle embedment process. J Vis Commun Image R 2016;40:574–88.
[8] Kamal AHM, Islam MM. Capacity improvement of reversible data hiding scheme through better prediction and double cycle embedding process. In: Proceedings of IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Kolkata, India; December 2015. 16-18.
[9] Kamal AHM, Islam MM. Enhancing the embedding payload by handling the affair of association and mapping of block pixels through prediction errors histogram. In: Proceedings of International Conference on Networking, Systems and Security (NSysS), BUET, Dhaka; January 2016. 5-8.
[10] Kamal AHM, Islam MM. Enhancing embedding capacity and stego image quality by employing multi predictors. J Inf Secur Appl. 2017;32:59–74.
[11] Ma Xiaoxiao, et al. High-fidelity reversible data hiding scheme based on multi-predictor sorting and selecting mechanism. J Vis Commun Image Represent 2015;28:71–82.
[12] Pan Z, et al. Reversible data hiding based on local histogram shifting with multilayer embedding. J Vis Commun Image R 2015;31:64–74.
[13] Ou, Bo, et al., "Reversible data hiding based on pde predictor.", J Syst Softw, 86.10 (2013): 2700–9
[14] Wang J, Ni J, Hu Y. An efficient reversible data hiding scheme using prediction and optimal side information selection. J Vis Commun Image Represent 2014;25.6:1425–31.
[15] Leung HY, Cheng LM, Liu F, Fu QK. Adaptive reversible data hiding based on block median preservation and modification of prediction errors. J Syst Softw 2013;86.8:2204–19.
[16] Habiba S, Kamal AHM, Islam MM. Enhancing the robustness of visual degradation based ham reversible data hiding. J Comput Sci 2016;12.2:88–97.
[17] Kamal AHM, Islam MM. An image distortion-based enhanced embedding scheme. Iran J Comput Sci 2018;1.3:175–86.
[18] Liao Xin, Shu Changwen. Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels. J Vis Commun Image Represent 2015;28:21–7.
[19] Ong SimYing, Wong Koksheik, Tanaka Kiyoshi. A scalable reversible data embedding method with progressive quality degradation functionality. Signal Process Image Commun 2014;29.1:135–49.
[20] Zhang Xinpeng, et al. Efficient reversible data hiding in encrypted images. J Vis Commun Image Represent 2014;25.2:322–8.
[21] Hong Wien. Adaptive reversible data hiding method based on error energy control and histogram shifting. Opt Commun 2012;285.2:101–8.
[22] BOSS: Bank of standardized stimuli, https://drive.google.com/drive/folders/0B3m1Sf0USgt8b1VET3NLcTVIc0U, last visited: 16 January 2017.
[23] CalTech101: Images of category of 101 of computer vision lab of the california institute of technology, usa, https://www.vision.caltech.edu/Image_Datasets/Caltech101/101_ObjectCategories.tar.gz, last visited: 16 January 2017.
[24] USC-SIPI Standard. Signal and image processing institute. USA: University of Southern California; January 2017 http://sipi.usc.edu/database/database.php?volume=misc last visited: 16.
[25] Ulutas M, Ulutas G, Vasif VN. Medical image security and epr hiding using Shamir's secret sharing scheme. J Syst Softw 2011;84.3:341–53.
[26] Wong K, Tanaka K. Scalable scrambling method using unified constructive permutation function. In: Proceedings of the IIEEJ Image Electronics and Visual Computing Workshop; 2010.
[27] Zhou Wang ACB. A universal image quality index. IEEE Signal Process Lett 2002;9:81–4.