EURASIP Journal on Information Security
a SpringerOpen Journal

RESEARCH Open Access

# Edge-based image steganography

Saiful Islam*, Mangat R Modi and Phalguni Gupta

**Abstract**

This paper proposes a novel steganography technique, where edges in the cover image have been used to embed messages. Amount of data to be embedded plays an important role on the selection of edges, i.e., the more the amount of data to be embedded, larger the use of weaker edges for embedding. Experimental results have shown that the proposed technique performs better or at least at par with the state-of-the-art steganography techniques but provides higher embedding capacity.

**Keywords:** Steganography; Steganalysis; Information hiding; Edge detection

## 1 Introduction

Steganography is an art of secure transmission of messages from a sender to a receiver. It should ensure that no one can reliably conclude on the secret communication between the sender and the receiver. To achieve such a secrecy, the message is hidden in some cover media which may not raise any suspicion on the possibility of carrying the secret message to the third party. Embedding introduces distortion in the cover medium. The embedding distortion in visual and statistical properties of the cover medium may lead steganographic detectability. The objective of any steganographic technique is to preserve these properties while embedding the message in the cover media.

Images are preferred medium for the current steganography techniques. Content adaptability, visual resilience, and smaller size of images make them good carrier to transmit secret messages over the internet. There exists a large number of image steganography techniques which are accompanied by various attacks on the steganography systems. Security of any steganography technique depends on the selection of pixels for embedding. Pixels in noisy and textured area are better choice for embedding because they are difficult to model. Pixels in edges can be seen as noisy pixels because their intensities are either higher or lower than their neighboring pixels due to sudden change in the coefficient gradient. Due to these sharp changes in the visual and statistical properties, edges are difficult to model in comparison to pixels in smoother area. Therefore, edges make a better option to hide secret data than any other region of an image where a small distortion is much more noticeable. Figure 1a is an image with 20% of pixels modified to produce distortion. The image has some smooth parts such as sky and some parts with high concentration of edges, such as trees and buildings. Some areas from both smoother part and high texture part are cropped and zoomed as shown in Figure 1b,c. It can be seen that the modified pixels in the smoother parts are clearly noticeable, whereas it is hard to detect these distortions in the high texture parts. In this paper, we have proposed a steganography technique which can hide the secret message only in the edges of the cover image. The proposed steganography technique is found to have excellent security against steganalysis attacks. The performance of the technique is analyzed by testing on 'break our steganography system' (BOSSbase) ver. 1.01 [1] and 'break our watermarking system' (BOWS2)[a] databases, each having 10,000 grey scale images.

The paper is organized as follows. Section 2 discusses some well-known steganographic techniques. Various observations which are used to propose the steganographic technique are discussed in Section 3. An efficient edge-based steganography technique has been presented in Section 4. Experimental results have been analyzed in Section 5. Conclusions are given in the last section.

## 2 Literature review

There exist several steganographic techniques to embed data securely in a carrier medium and tools to detect reliably the presence of any secret message in a steganogram.

*Correspondence: sislam@cse.iitk.ac.in
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, Kanpur 208016, India

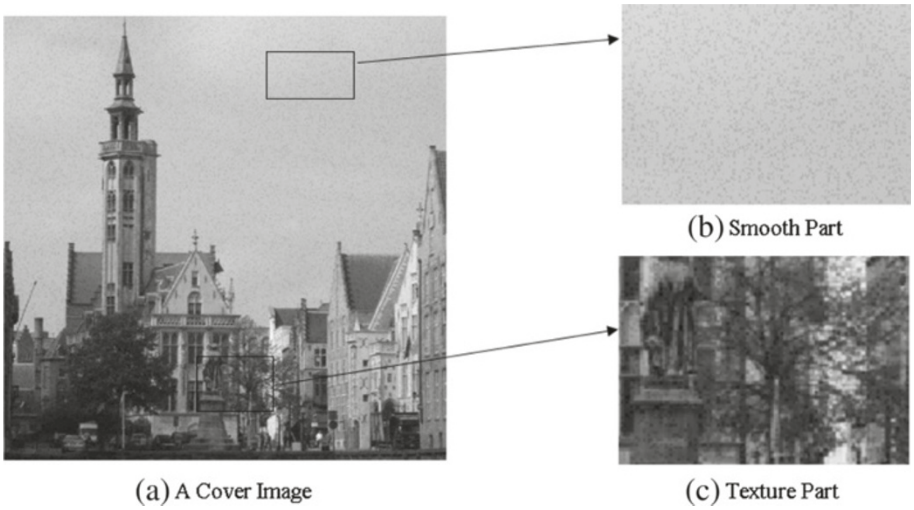

**Figure 1 Effect of embedding in an image. (a)** Cover image. **(b)** Smooth part. **(c)** Textured part.

Steganographic technique consists of embedding and extracting mechanism. Image-based steganographic techniques can be classified into two categories: spatial domain and frequency (transform) domain.

A secret message is generally considered as an encrypted data, where bits of encrypted message are embedded in pixels of the cover image. The trivial steganography technique is based on the least significant bit (LSB) substitution in which the LSB of the pixels is modified to embed the secret message. In the spatial domain, this type of techniques can be broadly classified into two categories: LSB replacement and LSB matching. In case of LSB replacement [2,3], the least significant bit of each pixel of the cover image is replaced by the next bit of the secret message to be embedded. In LSB matching [4], if there is a mismatch between least significant bit of a byte in the cover image and next bit of the secret message to be embedded, then embedding, in general, is done by increasing or decreasing randomly the content of the byte of the cover image by 1, except at the boundary values. In some techniques, the decision to increase or decrease the content of a byte is governed by the score of the distortion function [5]. Embedding in two least significant bits is an extension of LSB replacement. There are multiple ways to embed data by flipping the least and the second least bits of a cover image [6].

In case of transform domain, the LSB-based embedding is done by modifying the LSB of non-zero DCT coefficients of a cover image. There exist several ways to embed data in transform domain such as modification of quantization table, heuristic based, utilizing non-shared selection, and side information at sender side [7].

Steganalysis tools track the distortion caused during the data embedding to detect the presence of the secret message in an image. These tools are classified as visual, structural, and non-structural [8,9]. Visual steganalysis attacks analyze images for some distortions which are visible to human vision system. The distortions could be visible in stego image or in LSB plane extracted from the stego image. Structural attacks analyze structural properties of an image to find any anomaly which are introduced by steganography. Structural detectors such as histogram attack [10], sample pair analysis (SPA) [11], RS method [12], and weighted stego [13] can reliably detect presence of stego data and even estimate message length. Non-structural detectors use feature extractors to model cover image and to compute distortion between the cover and the stego image to detect embedding. A classifier is trained by the feature set from large number of stego and cover images. During training, the classifier learns the differences in features, and this learning is used to classify a fresh image into stego or clean image. Non-structural detectors such as subtractive pixel adjacency matrix (SPAM) [14] and spatial-rich model (SRM) [15] claim better probability of detection of embedding in a stego image. Features based on steganalysis techniques use support vector machine (SVM) or ensemble classifiers [16] for supervised learning. SVM is not suitable for any high-dimension feature vector, while this is not the case with ensemble classifier but its performance is comparable to SVM.

Most of the current steganography techniques are based on model-preserving principles. These techniques are designed by finding a model for cover images, and embedding modifications are done in such a way that this model is preserved. Highly undetectable stego (HUGO) [5], ASO [17], universal wavelet relative distortion (UNIWARD) [18], and maximum mean discrepancy (MMD) [19] are designed on this principle. HUGO preserves features used by SPAM for steganalysis, thus preserving features space model. Similarly, UNIWARD preserves a wavelet-based model, while MMD preserves parametric-based model. Generally, these techniques embed message by minimizing a defined embedding distortion function heuristically. But, in [20], a non-heuristic distortion function is used to preserve the Kullback Leibler distance.

In [21], an embedding technique, known as pixel value difference technique (PVD) has been proposed. In this technique, the image is divided into non-overlapping blocks of adjacent pixels which are randomly selected, and data is embedded into each of its pixels. The amount of data embedded, i.e., the number of last significant bits used, is directly proportional to the differences in the intensities of adjacent pixels. This uneven embedding in PVD leads to unusual steps in the histogram of pixel difference in the stego image. An improved technique (IPVD), proposed in [22], has exploited this vulnerability. Adaptive edge LSB technique (AE-LSB) [23] has also removed this uneven pixel difference by introducing a readjusting phase and has provided better capacity. All these techniques are edge adaptive in a way that they can embed more data where pixel difference is high but they have one fundamental limitation. These techniques consider pixel pair at random, rather than selecting on the basis of higher differences. So, they may end up by embedding data at random places in the image and by distorting the texture in LSB plane of the image. Performance of these techniques are found to be poor [24].

In hiding behind corners (HBC) [25][b] technique, corner pixels are used to contain hidden data. Data is embedded by using simple LSB substitution. Such embedding leads to many structural asymmetries and could easily be detected by structural steganalysis tools like chi-square [10], sample pair analysis (SP) [26], and weighted stego (WS) [27]. Thus, the HBC technique which maintains texture in LSB plane, offers poor security.

Edge adaptive image steganography (EALMR) [24] technique is based on LSB matching revisited (LSBMR) [3] technique which alleviates some of the above said limitations. EALMR calculates the difference between two adjacent pixels. If this difference is greater than a predefined threshold, then both pixels are marked as edge pixels, and one bit of data is hidden in each of them using LSBMR. This technique has some limitations. Difference of intensities of adjacent pixels may not be an edge point; any such technique may embed data in smoother parts even though there are some unused prominent edges. So, any well-known edge detection algorithm can be used to find edge pixels and to hide data in the detected edges. Further, since EALMR compares a pixel with its adjacent pixel, it can find edges only in one direction. To overcome this limitation, an image can be divided into some non-overlapping but equal size blocks, and each block is rotated in the range of set {0°, 90°, 180°, 270°} to see edge pixels in more than one direction inside a given block. But, poor edge selection results in detection by steganalysis tools like targeted attack [28] and blind attacks SPAM [14] and SRM [15].

In [5], HUGO steganographic technique is presented. Its design is derived from the image model obtained from the feature set of SPAM steganalyzer. It is based on the minimum-embedding impact principle, where embedding is done in such a way that the distortion in a stego image is minimum. It preserves a model utilized by SPAM steganalyzer to derive steganalytic features in such a way that it does not over-fit to a SPAM feature set. Dimensionality of the feature set has been tremendously enhanced so that the technique is not detectable by minor modification in SPAM steganalyzer. Instead of using Markov transition matrix to compute SPAM features, co-occurrence matrix is used to derive those features. But, it may have minor degradation in performance. Detectable parts of the model are identified by Fisher linear discriminant (FLD criteria) [29]. It rates individual features' importance for embedding changes. The parts of the model not vulnerable to embedding changes are identified using criteria optimized in FLD. In [30], it is shown that HUGO is vulnerable against steganalysis that uses other models drawn from different domains.

In [31], embedding distortion cost is computed through directional residual obtained using Daubechies wavelet filter bank [32]. The objective is to limit the embedding changes to those parts of the cover image that are difficult to model in multiple directions. Embedding is done in textures or noisy parts and avoiding smooth regions and clean edges of empirical cover images. Distortion function, called as UNIWARD [31], is used to compute delectability map. Syndrome trellis code (STC) [33] and detectability map are used to embed payload while minimizing the embedding distortion. The same distortion design technique can be used for spatial and transform domains.

## 3 Some observations

This section discusses some observations that are used to design an edge-based steganography technique. The edges are difficult to model, and the pixels belonging to each selected edge are considered as noisy pixels for embedding. In Figure 2, it is seen that embedding in edge pixels leads to changes in edges of the stego image. Consider the cover image shown in Figure 2a. The edges in the cover image for maximum possible embedding capacity are shown in Figure 2b, and corresponding edges in the stego image are depicted in Figure 2c. Finally, Figure 2d shows the locations where edges in stego and cover image

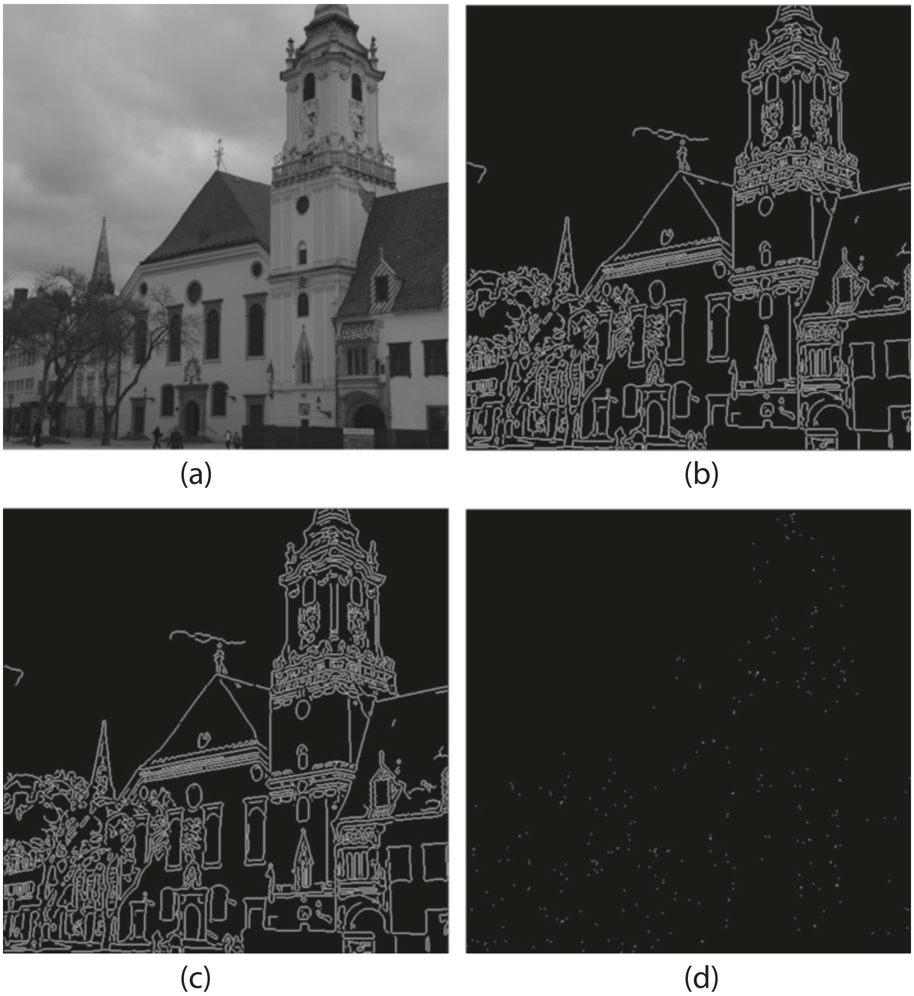

**Figure 2 Change in the number of edge pixels due to embedding. (a)** Cover image. **(b)** Edge pixels in a cover image. **(c)** Edge pixels in a stego image. **(d)** Edge pixel mismatch.

do not match. As a result, it makes it impossible to retrieve the embedded message from the stego image. But, the secret message must be extracted from the stego image. Therefore, just by using any edge detection algorithm to detect edges and embedding in those locations may not be sufficient for designing an edge-based steganography technique.

### 3.1 Masking cover image

In order to keep no changes in edges before and after embedding, the LSBs of the cover image are masked, and edge detectors are applied on the masked cover images. Since LSB replacement does not modify any bit other than LSB, a pixel of a cover image, the edges in cover and stego images remain identical as shown in Figure 3. It has been observed that the number of pixels belonging to edges does not change much by masking LSB or the least two significant bits (2LSB).

Table 1 lists the difference in the number of pixels belonging to edges between cover image and its modified images by masking 2LSB for BOSSbase database ver. 1.01 of 10,000 natural images. It can be seen that the average difference for different edge detectors and masking 2LSB is limited to less than 2%. However, there is a outlier case, shown in Figure 4, where the difference is 61%. It can be noted that for both databases, the number of pixels belonging to edges are increased after masking 2LSB. Hence, masking at least two significant bits does not effect the edges in the cover image for most of the cases.

### 3.2 Embedding in LSB or 2LSB

Most of the steganographic techniques embed data in LSB of pixels in the cover image pixel. Embedding is done by either LSB replacement or LSB matching. LSB replacement is detected by most of the structural detectors, but

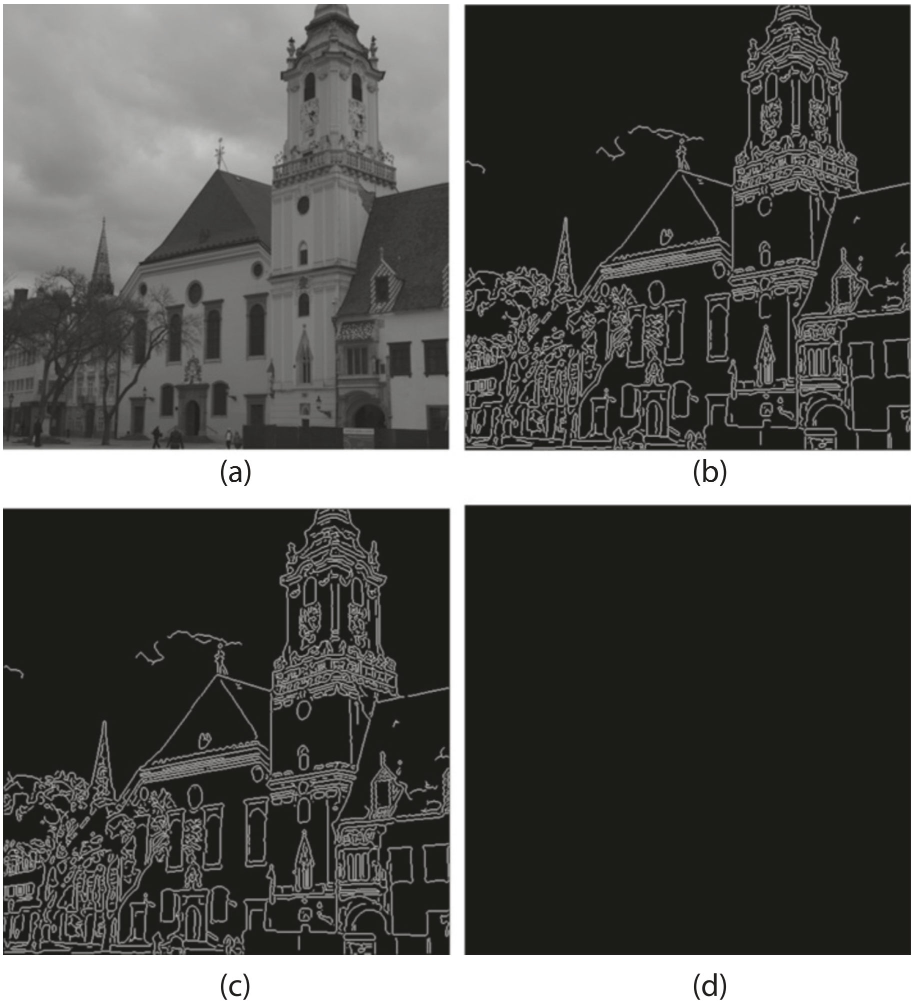

**Figure 3 Change in the number of edge pixels due to embedding. (a)** Cover image. **(b)** Edge pixels in a cover (masked). **(c)** Edge pixels in a stego (masked). **(d)** No edge pixels mismatch.

LSB matching is reliably detected through non-structural detector SPAM and SRM. Hence, if one embeds in the LSB plane, then there is a high probability of detection of presence of the message. To overcome these structural and non-structural detectors, embedding is done in 2LSB plane of the cover image. Embedding in 2LSB plane violates the basic assumption of structural detectors, and it has been observed that even SPAM and SRM are less accurate in detecting the presence of the message for less amount of data embedding in comparison to LSB replacement. It is shown in [6,34] that embedding in 2LSB plane is preferable to embed in LSB plane.

## 4 Proposed technique

This section proposes a new steganography technique which hides secret messages in the edges of the carrier

**Table 1 Average difference in number of edge pixels between image and its (2LSB) masked image**

| Algorithm | Total pixels | Edge pixels (average) | Edge pixels (%) | Edge pixels (masked) | Difference (average) | Difference (%) |
|---|---|---|---|---|---|---|
| Canny | 262,144 | 23,818 | 9.1 | 24,201 | 383 | 1.61 |
| Sobel | 262,144 | 8,451 | 3.2 | 8,458 | 7 | 0.09 |
| Prewitt | 262,144 | 8,407 | 3.2 | 8,415 | 8 | 0.10 |
| LOG | 262,144 | 18,220 | 6.9 | 18,357 | 137 | 0.80 |

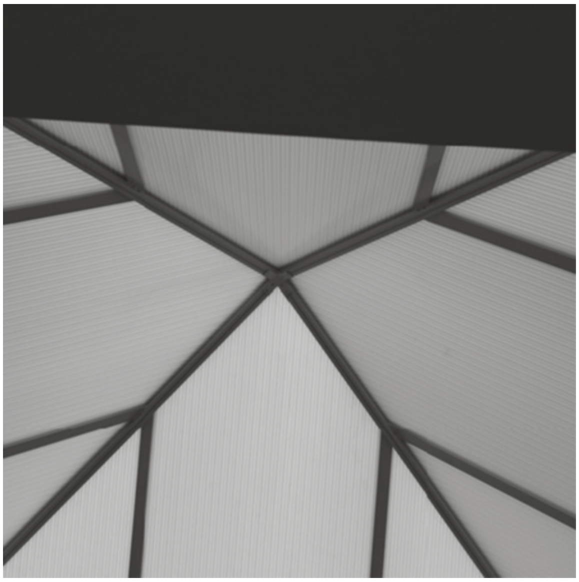

**Figure 4 A cover image of an outlier case of edge difference.**

image. It is an extension of edge embedding in color image [35]. To get true edges, Canny edge detection technique [36] has been used. The selection of edges for embedding is dependent on the length of payload and the image. As the payload size increases, a weak threshold for the selection of edges is used so that more edges can be selected to accommodate the increased amount of data. For a given payload, the sharpest possible edges are selected to embed the message. The flow graph shown in Figure 5 shows the proposed steganography technique which consists of two primary tasks: threshold selection and embedding. Threshold selection is to find Canny high threshold $t_\mathrm{h}$ so that sufficient number of edges are selected to embed the given payload in a cover image, while embedding is done by computing edge-map based on threshold. Payload is embedded in a cover image in a random order based on the stego key and the edge map.

### 4.1 Threshold selection

Canny edge detection algorithm returns [36] edges in a image on the basis of three parameters, namely, high threshold ($t_\mathrm{h}$), lower threshold ($t_\mathrm{l}$), and width of Gaussian kernel. Threshold $t_\mathrm{h}$ is used to identify strong edges, where $t_\mathrm{l}$ helps to identify weaker edges. The $t_\mathrm{h}$ value is dynamically adjusted on the basis of message size in such a way that enough number of edges in the cover image are selected to embed the secret message. Experimentally, the $t_\mathrm{l}$ value is set to $0.4 \times t_\mathrm{h}$. The effect of change in the threshold value for edge detection is shown in Figure 6. The sharper edge means sharp change in visual and statistical properties in the image which make the detection of hidden data tougher. Canny edge detector's sensitivity to noise is controlled by width of the kernel. Increase of the kernel width decreases detector's noise sensitivity and vice versa. Initially, the width of the kernel is taken as constant to obtain the threshold value. Initial guess of the threshold is obtained through Algorithm 1. Later, fine tuning of the threshold and width of the Gaussian kernel are done to improve the embedding algorithm. Figure 7 shows the effect of width of kernel $w$ on thresholds $t_\mathrm{h}$ and $t_\mathrm{l}$ and number of pixels selected for embedding.

It is to be noted from Table 2 that for each image, different sets of thresholds are obtained to fix the width of the kernel. There is no way to obtain the best value of width of the kernel, but it has been observed that high value of $t_\mathrm{h}$ and low value of $w$ are better options for embedding. For experimental purpose, fine tuning is done by increasing the threshold value and decreasing width of the Gaussian kernel. The results are evaluated for fixed values of $w$ and embedding rates. Sets of stego images are evaluated for security, and the best result among all cases is reported. Threshold selection tries to avoid clean edges by fixing $t_\mathrm{l} = 0.4 \times t_\mathrm{h}$ and by reducing the width of Gaussian kernel.

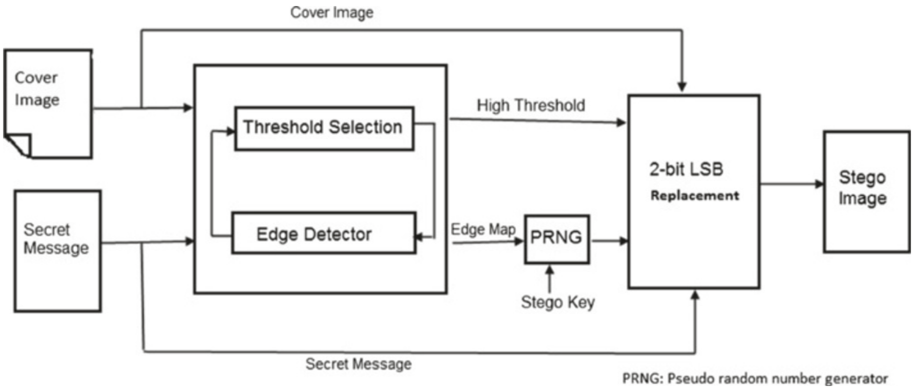

**Figure 5 The proposed steganography technique.**

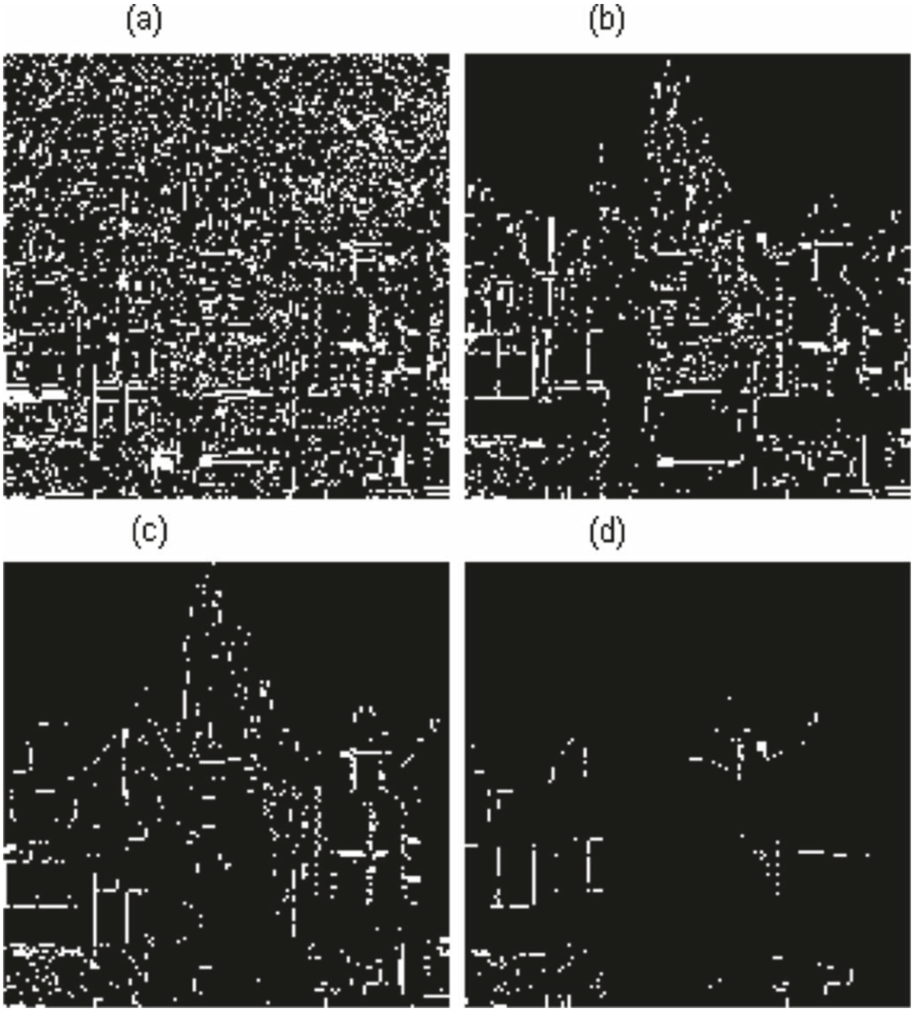

**Figure 6 Effect of threshold on edge detection. (a)** Threshold 0.00. **(b)** Threshold 0.25. **(c)** Threshold 0.50. **(d)** Threshold 0.75.

To facilitate the extraction of message at the receiver end, the length of the secret message is pre-fixed to the message to form an augmented message. The structure of the augmented message is shown in Figure 8. The message size field of the augmented message is of fixed length ($C$ bits). The augmented message is a binary string with the assumption that $C$ bits are sufficient to store the message size information.

The threshold value is computed at the time of embedding, and it has to be sent to the receiver separately. The threshold value and width of the kernel are stored in IEEE 754 floating point half precision format. They require only 16 bits and are embedded in non-edge pixels of the cover image.

Algorithm 1 finds the suitable high threshold value for the Canny edge detector. Initially $t_{\min}$ is set to 0 and $t_{\max}$ equal to 1. The high threshold value $t_{\mathrm{h}}$ lies between these two values. Let the number of pixels in edges returned by Canny edge detector for the given threshold which is median between $t_{\min}$ and $t_{\max}$ be $n_{\mathrm{e}}$. Binary search is used to find the threshold value. It is quite possible that number of pixels $n_{\mathrm{e}}$ belonging to edges may not be exactly same as length of the secret message $N$. If this condition occurs, binary search cannot return the threshold value. To alleviate this problem, the terminating condition of the search is modified so that it returns the number of pixels greater than or equal to $N$, and *limit* is used to set upper bound on $n_{\mathrm{e}}$. It has been found that *limit* of 1% is sufficient for BOSSbase ver. 1.01 and BOW2 database. If $n_{\mathrm{e}}$ is less than the required number of edge pixels $N$, then $t_{\mathrm{h}}$ has to be greater than the median threshold, $t_{\min}$ is set to the median threshold, and the process is repeated. Similarly, if $n_{\mathrm{e}}$ is greater than $(N + 0.01N)$, the median threshold is set to $t_{\max}$, and the process is repeated until the difference between the number of edge pixels, $n_{\mathrm{e}}$ and $N$, is less than the *limit*.

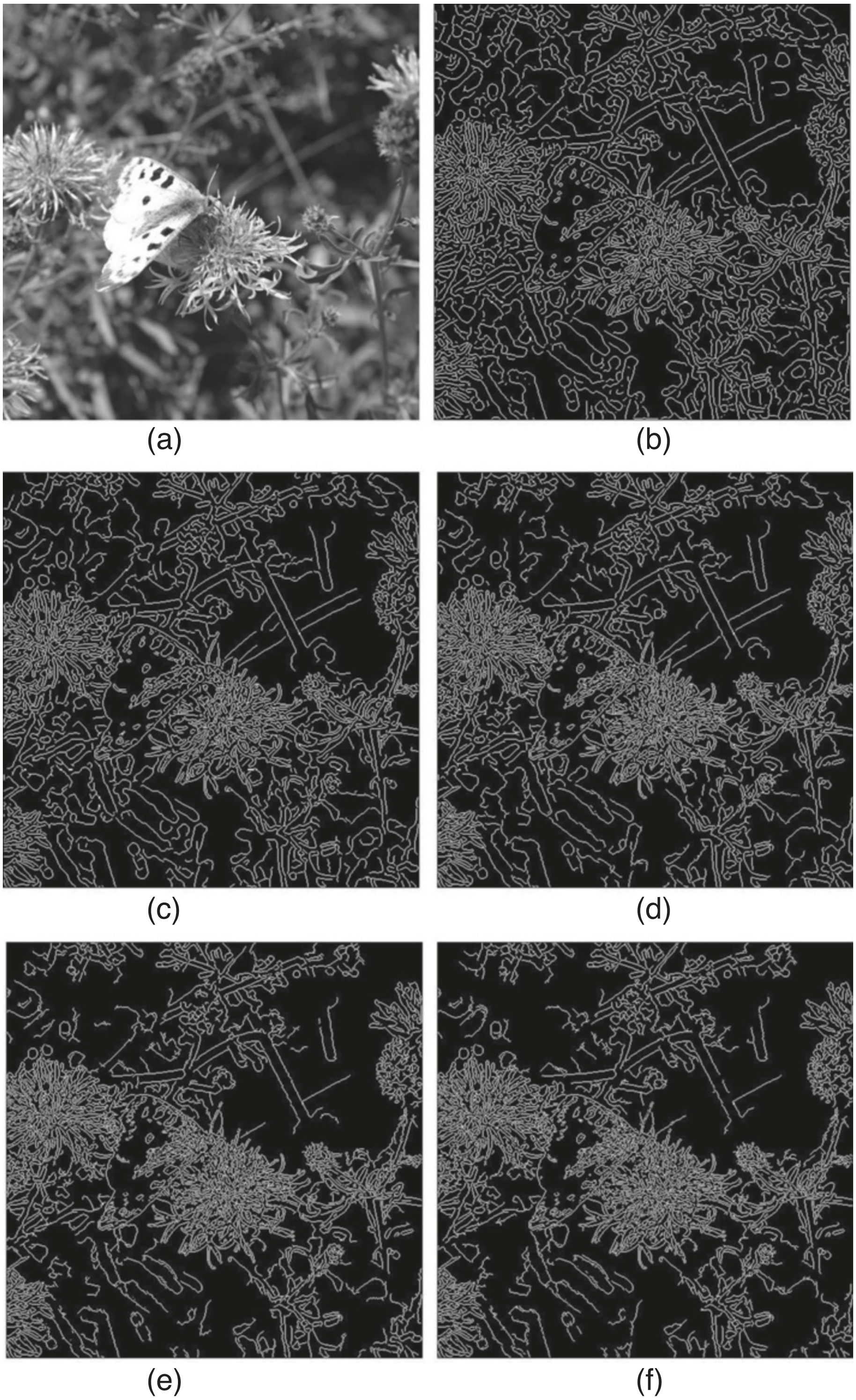

**Figure 7 Effect of width of the kernel on edge pixels for an embedding rate of 0.10 bits per pixel (bpp). (a)** Cover image. **(b)** Relative width = 1.0. **(c)** Relative width = 0.8. **(d)** Relative width = 0.6. **(e)** Relative width = 0.4. **(f)** Relative width = 0.2.

### 4.2 The embedding

Embedding in the cover image is done by least two-significant-bit substitutions (2LSB). It means that each of the least two significant bits in the pixel intensity holds one bit of message. Bits of a pixel are flipped whenever they are not equal to the message bits. Only six most

**Table 2 Effect of width of kernel on edge pixels**

| High threshold ($t_h$) | Low threshold ($t_l$) | Width ($w$) | Pixels required (range in bits) | Embedding capacity (bits) |
|---|---|---|---|---|
| 0.070 | 0.0280 | 1.0 | 26,234 to 28,835 | 28,801 |
| 0.105 | 0.0420 | 0.8 | 26,234 to 28,835 | 28,598 |
| 0.105 | 0.0420 | 0.6 | 26,234 to 28,835 | 28,506 |
| 0.095 | 0.0380 | 0.4 | 26,234 to 28,835 | 28,798 |
| 0.095 | 0.0380 | 0.2 | 26,234 to 28,835 | 28,422 |

**Algorithm 1:** getThreshold($I, N, w$)

```
Data: I: Image, N: Length of augmented message to be
      embedded, w: width of the Gaussian Kernel
Result: threshold: $t_h$ for Canny to get N pixels
// limit is set to 1% of the message length
// no. of edge pixels, $n_e \leq N + 0.01 \times N$ and $n_e \geq N$
// $n_e$ = number of edge pixels in I, when Canny edge detector is used on I with high threshold $t_h$ and low threshold $t_l = 0.4 * t_h$ and width w

limit ⟵ $0.01 \times N$;
$t_{max}$ ⟵ 1;
$t_{min}$ ⟵ 0;
set ⟵ false;
repeat
    $t_h$ ⟵ $\lfloor (t_{max} + t_{min})/2 \rfloor$;
    $n_e$ ⟵ getEdgePixelCount(Canny($I, t_h, t_l, w$));
    // it returns the number of pixels in the edges obtained through Canny edge detector
    diff ⟵ $n_e - N$;
    if diff > limit then
        $t_{min}$ ⟵ $t_h$;
    end
    else if diff < 0 then
        $t_{max}$ ⟵ $t_h$;
    end
    else
        set ⟵ false;
    end
until set = true;
return $t_h$
```

significant bits participate in edge detection to form an edge map of the gray scale cover image.

In Algorithm 2, two least significant bits of the cover image are masked to form an edge map ($e$). Threshold is computed using the masked cover image and length of the augmented message. The number of edge pixels used to embed the augmented message is half of the augmented message bits because each edge pixel carries two bits of the augmented message. The edge map $e$ obtained through the Canny edge detector is randomly arranged using stego key $P$ by calling *randomPermute(e,P)*. It ensures that only the intended users can extract data from the stego image. The secret message $M$ is embedded in the randomly permuted $S$ using edge map $e$ by modifying the least two bits of pixel $S_{x,y}$ to the corresponding two consecutive message bits $M_{\text{index}+1}$ and $M_{\text{index}}$ . The threshold and width are embedded in non-edge pixels of the stego image. Non-edge pixel map $e'$ is obtained by taking complement of $e$ for minimum values of $t_h$ and $w$. The threshold and width are embedded in the first 32 bits of $S$ corresponding to $e'$. Image $S$ is reshuffled to get the stego image.

### 4.3 Extraction

It is the process to retrieve the augmented message from the given stego image. The threshold value and width are extracted from the non-edge pixels of the stego image. Stego key has been used as the seed to permute the set of edge pixels. Extraction is similar to the embedding process. The least two significants bits of the stego image $S$ is masked, and edge map $e$ is computed. It is permuted using stego key $P$ to retrieve the message in the same order as it has been embedded. The value corresponding to the least

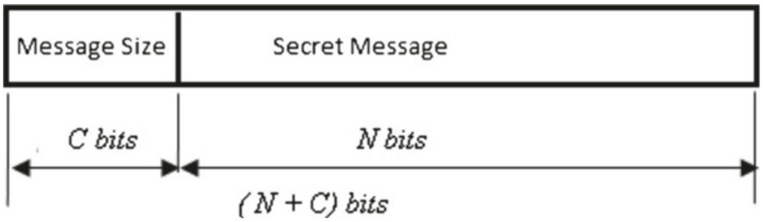

**Figure 8** Augmented message to be embedded.

**Algorithm 2:** embed($I$, $M$, $P$, $w$)

```
Data: $I$: Image, $M$: Augmented message in binary, $P$: Stego
      key, $w$: width of the Gaussian kernel
Result: $S$: Stego image
$S \longleftarrow I$;
// Mask 2LSB and find edges
$I \longleftarrow$ bitand($I$,252);
$L \longleftarrow |M|$;
// Length of the augmented message
// Compute Threshold
$t_h \longleftarrow$ getThreshold($I, L, w$);
// Obtain e: e is edge map obtained by
   calling Canny edge detector algorithm
   with high threshold $t_h$ and low
   threshold $t_l = 0.4 * t_h$ and width $w$
$e \longleftarrow$ Canny($I$, $t_h$, $t_l$, $w$);
// Shuffle eand S using Stego key $P$
$e \longleftarrow$ randomPermute($e$,$P$);
$S \longleftarrow$ randomPermute($S$,$P$);
// e is the set of edge pixels
index $\longleftarrow$ 0;
for each edge pixel i in e do
    $S_{x,y}$ = bitand($S_{x,y}$,252); // x,y are
       co-ordinates of pixel $i$
    $S_{x,y} = S_{x,y} + 2*M_{\text{index}+1} + M_{\text{index}}$;
    index $\longleftarrow$ index + 2;
end
// Embed threshold and width in non-edge
   pixels of $S$
$e \longleftarrow$ Canny($I$, 0, 0, 0.1); // Pixels in e are
   maximum number of edge pixels for a
   given image
$e' \longleftarrow$ complement(e); // Pixels in $e'$ are
   non-edge pixels
for i = 1: 16 in e' do
    $S_{x,y}$ = bitand($S_{x,y}$,254); // $x,y$ are co-ordinates
       of pixel$i$
    $S_{x,y} = S_{x,y} + t_{h(i)}$;
    // $t_h$ is represented in 16 bits IEEE
       754 floating point half precision
end
for i = 17: 32 in e' do
    $S_{x,y}$ = bitand($S_{x,y}$,254); // $x,y$ are co-ordinates
       of pixel $i$
    $S_{x,y} = S_{x,y} + w_{(i-16)}$;
    // $w$ is represented in 16 bits IEEE
       754 floating point half precision
end
$S \longleftarrow$ randomPermute($S$,$P$); // Reshuffle $S$ to
   get Stego Image: $S$
return $S$;
```

**Algorithm 3:** decode: retrieve secret message

```
Data: $I$: stego image, $T$: Threshold, $P$: stego key, $w$: Kernel
      width
Result: $M$: Secret message
$S \longleftarrow I$;
// Mask least 2 bits and find edges
$S' \longleftarrow$ bitand($S$,252);
// Obtain e: e is edge map obtained by
   calling Canny edge detector algorithm
   with high threshold $t_h$ and low
   threshold $t_l = 0.4 * t_h$ and width $w$
$t_h \longleftarrow T$;
$t_l \longleftarrow 0.4 * t_h$;
$e \longleftarrow$ Canny($S'$,$t_h$, $t_l$, $w$);
$e \longleftarrow$ randomPermute($e$,$P$);
// Shuffle $S$ to get order of embedding
$S \longleftarrow$ randomPermute($S, P$);
index $\longleftarrow$ 0;
for each edge pixel i in e do
    val $\longleftarrow$ bitand($S_{x,y}$,3); // $x,y$ are co-ordinates
       of pixel i
    $M_{\text{index}+1} \longleftarrow val \bmod 2$;
    val $\longleftarrow$ val/2;
    $M_{\text{index}}$ = val;
    index $\longleftarrow$ index + 2;
end
// extract first $C$ bits to get message
   size
msg_size $\longleftarrow$ $M$[1:$C$];
$M \longleftarrow M[C + 1 :$ msg_size$]$;
return ($M$);
```

two significant bits of each edge pixel is extracted to *val*. Two consecutive bits $M_{\text{index}+1}$ and $M_{\text{index}}$ of payload are retrieved from *val* for each edge pixel belonging to *e*. This retrieved payload consists of message size, actual message, and few extra bits. Message size *msg_size* is extracted from the first *C* bits of the payload which is used to retrieve the actual message *M*[*C+1:msg_size*]. Extra bits beyond *msg_size* are discarded, and the secret message *M* is returned. Algorithm 3 delineates the extraction module.

## 5 Experimental results

The proposed technique has been tested on BOSSbase database ver. 1.01 and BOWS2 database. Each database contains 10,000 8-bit gray scale images with size of 512 × 512. Images of BOSSbase taken from eight different cameras, resized and uncompressed. Secret message is randomly generated by a pseudo random number generator (PRNG) to simulate encryption of the secret message. For experimental purpose, payload is taken to be 10% bits per pixel (bpp) of the cover image to show the effectiveness of the proposed technique. In both databases, the total number of pixels belonging to edges is found to be less than 10%. Further, the technique is also analyzed for variable payload and for the best threshold and width of the Gaussian filter. It is compared with LSBM, existing edge adaptive techniques HBC and EALMR, and minimizing distortion-based techniques HUGO and S-UNIWARD. The steganographic security is evaluated against visual, structural, and blind steganalysis attacks.

### 5.1 Visual attacks

All well-known techniques do not create any visible mark on the stego image. However, when the LSB plane is filtered to remove the significant part of the cover image, the difference becomes obvious. The technique like LSBM embeds data, irrespective to the nature and structure of image damaged texture in the LSB plane. One of the reasons to use edge-based steganography is to preserve the texture in the LSB plane. Texture in LSB plane can be seen in Figure 9b which has some white and black patches; these are the clusters of pixels having same LSB value. This property is more prominent in smoother areas of the image like bright sky or dark shadow. So, any change in the smoother part of the image may change LSB value of the pixels in these clusters. This is visible in Figure 9c, which is a stego image generated by LSBM, corresponding to the area marked in Figure 9b. Here, LSBM writes some message bits, causing some black pixels to appear. On inspection, these black pixels on the white patch may raise suspicion. On the other hand, all other edge-based techniques in Figure 9 have not caused any noticeable change. It can be concluded that the edge-based steganography resists visual attacks.

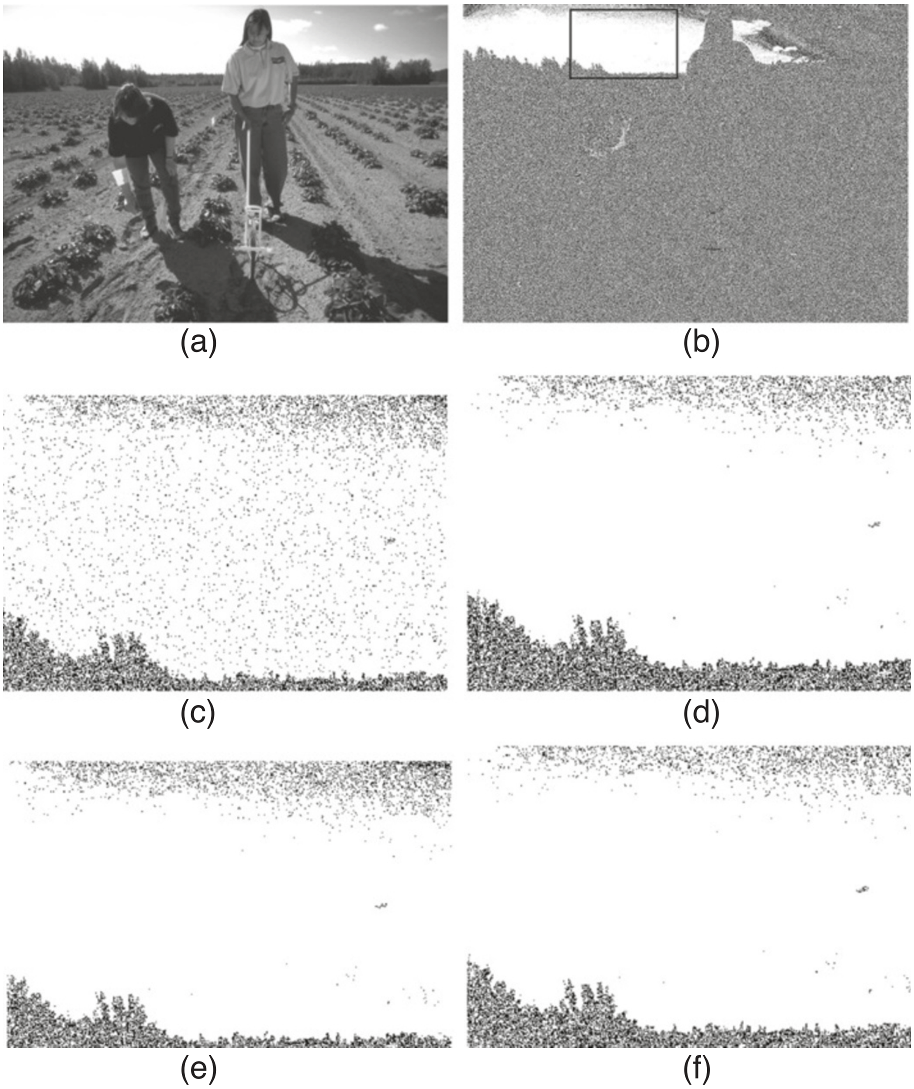

**Figure 9 LSB plane of cover image with 10% embedding and smooth area (marked) is cropped from stego images. (a)** Cover image. **(b)** Cover image LSB plane. **(c)** LSBM. **(d)** EALMR. **(e)** HBC. **(f)** Proposed technique.

### 5.2 Structural attacks

Embedding data in an image leads to statistical modification in the structure of cover image. Any such modification in a cover image can be observed by first- and second-order statistics. SP [26] analysis and WS [27] are two popular structural attacks. Both SP and WS estimate the length of the embedded message by giving the percentage of pixels which may hold data.

Table 3 lists the result of SP and WS for various steganography techniques. It can be noted that the relative message length for HBC lies close to 10%, but that for LSBM, EALMR, and the proposed technique is 0.2%, 0.06%, and 2.6%, respectively. For better understanding, SP and WS are executed on the original databases to get the mean value of relative message length. LSBM, EALMR, and the proposed technique are found to be close to the mean value of the natural images. One possible reason for these results could be the use of LSB matching and 2LSB-based embedding which do not lead to asymmetry in pixels intensity. Therefore, the relative message length for any edged-based technique does not raise any suspicion.

**Table 3 SP and WS relative message length for various steganography techniques (estimation results)**

| Database | Attacks | LSBM | HBC | EALMR | Cover image | Proposed |
|---|---|---|---|---|---|---|
| BOWS2 | SP | 0.20 | 15.6 | 0.06 | 2.40 | 2.60 |
| | WS | 0.12 | 8.34 | 0.01 | 0.84 | 1.30 |
| BOSSbase ver. 1.01 | SP | 0.32 | 9.60 | 0.09 | 0.01 | 0.07 |
| | WS | 0.10 | 7.20 | 0.06 | 0.00 | 0.05 |

### 5.3 Blind steganalysis using feature extraction

Edge-based steganographic techniques, minimizing embedding distortion-based techniques, and the proposed technique are analyzed using SPAM and SRM.

#### 5.3.1 Steganalysis by Subtractive Pixel Adjacency Matrix (SPAM)

Analysis of these techniques is performed by taking feature sets from their respective stego images and natural images. These features are used to train SVM to learn the difference in features caused by steganography [37]. Testing is done in fivefold cross-validation. For each technique, cover and corresponding stego sets of images are divided into five parts. SVM is trained for four sets of randomly selected sample images, and the results have been validated for the remaining set. This process is repeated five times, and the average of all the tests is reported as the final result. The results from SVM may vary greatly with the values of two parameters: penalization cost ($C$) and gamma ($\gamma$). However, there are few rules to determine optimal values of these parameters. High or low values of both false positive ($F_p$) and false negative ($F_n$) indicate that the classifier is biased towards one class. For a good classifier, average value of $F_p$ and $F_n$ should be low. So, the values of $C$ and $\gamma$ are adjusted to achieve minimum average value of $F_p$ and $F_n$. In [38], fivefold cross-validation is used with the multiplicative grid

$$C \in \{0.001, 0.01, \ldots 10000\}$$
$$\gamma \in \left\{2^i \mid i \in \{-d-3, \ldots, -d+3\}\right\},$$

where $d = \log_2(x)$, $x$ is the number of features in the subset. But, the range of $\gamma$ has been extended as it has been observed that the accuracy of SPAM increases for the larger value of $\gamma$. Hence, for experimental purpose, the following multiplicative grid is used:

$$C \in \{0.001, 0.01, \ldots 10000\}$$
$$\gamma \in \{0, 0.01, 0.1, 1, 2, 4, 8, 16\}.$$

Table 4 shows that LSBM is detected with an accuracy of 93.0%. HBC and EALMR both do not use any well-known edge detection algorithm; therefore, they are easily detected by SPAM and have accuracy rate of 89.6% and 70.8%, respectively. The maximum detection accuracy achieved by the proposed technique is 51.1%, and it can be attributed to the selection of noisy pixels through true edge detection [36]. It can be noted that accuracy of 50% is like a random guess about cover and stego images. This means that features extracted by SPAM have failed to produce any considerable difference between stego and natural images for the proposed technique.

**Table 4 SPAM accuracy against edge embedding algorithms**

| Algorithm | Accuracy |
|---|---|
| LSBM | 93.0% |
| HBA | 89.6% |
| EALMR | 70.8% |
| Proposed | 51.1% |

#### 5.3.2 Steganalysis by spatial-rich model

SRM consists of 39 symmetrized sub-models quantized with three different quantization factors with a total dimension of 34671. Due to its large dimensionality, it is implemented using machine learning tool, ensemble classifier [16] which consists of many base learners such as FLD [29]. Each of them is trained on a set of a cover and stego images. The accuracy of the model is evaluated using the ensemble's unbiased estimate of the testing error known as the 'out-of-bag' error, $E_{\mathrm{OOB}}$. It is an accurate estimate of the testing error.

This subsection presents the results of the tests conducted for relative payloads 0.05, 0.10, and 0.20 bpp. It has been observed in Table 1 that for BOSSbase database, the number of pixels belonging to edges is less than 10%. The number of pixels belonging to edge can be increased by decreasing threshold and kernel width. This increase in the number of pixels is, on average, limited to 34.6% of cover images of the database. Relative payloads of 0.30, 0.40, and 0.50 bpp are not considered in the database because it is not possible to embed payload of 0.30 bpp in large number of images by the proposed technique. The number of cases for which the payload is more than the number of pixels belonging to edges increases considerably for embedding rate greater than 0.30 bpp. The simulation results of the proposed technique is compared with those of HUGO and S-UNIWARD [18]. The results are obtained through fine tuning of threshold and width of the Gaussian kernel and embedding using LSBMR. LSBMR is used, instead of 2LSB, because HUGO and S-UNIWARD embed at most 1 bit per pixel. The HUGO simulator with default settings is used to create the stego images. Similarly, the sets of stego images are obtained from the S-UNIWARD simulator. Figure 10 presents $E_{\mathrm{OOB}}$ errors of the proposed technique, HUGO and S-UNIWARD. It is seen that the proposed technique outperforms HUGO and performs equivalent to the S-UNIWARD for embedding rate up to 20%. Embedding rate beyond 20% cannot correctly be evaluated because there are large number of outlier cases as shown in Table 5. For relative payload of 30%, there are 1,331 cover images having less than 30% of total pixels belonging to edge. Therefore, embedding is not possible for the given payload in these images.

**Table 5 Number of outlier cases for the proposed technique**

| Bits per pixel | Outliers |
|---|---|
| 0.05 | 2 |
| 0.10 | 2 |
| 0.20 | 25 |
| 0.30 | 1,331 |

## 6 Conclusions

In this paper, a new technique for steganography in gray scale images has been proposed. Data is hidden at the edges of the cover image, and the edges are dynamically selected based on the length of the message. The proposed technique can resist visual, structural, and non-structural attacks better than the existing edge-based techniques. HBC is detected by structural detectors due to anomalies created by LSB substitution. These anomalies are well resisted by LSBM, but it does not discriminate between smooth areas and the edges in an image causing some distortion in LSB plane of stego image. EALMR is resistant to structural attacks because it uses LSBMR for embedding. It fails to discriminate between prominent edges and smothered area for a given threshold. Hence, there is a possibility of embedding in smoother parts of image. The proposed technique uses two-bit LSB substitution for embedding, and as a result, it decreases the number of pixels to be distorted. Modification of two bits of the selected pixels leads to significant change in pixel intensity, but this change does not lead to detectability due to sharp difference in intensity of edge and non-edge pixels. Hence, embedding in edges does not produce any visual distortion in stego images. The performance of the proposed technique is also found to be better than that of HUGO for embedding rate less than 10% bpp and slightly better than that of S-UNIWARD for embedding rate of 5%. An embedding rate greater than 10% bpp leads to embedding payload in weak edges. It tries to avoid clean edges by fixing $t_{\mathrm{l}} = 0.4 \times t_{\mathrm{h}}$ and reducing width of Gaussian kernel, but there is no rule of thumb to accurately discriminate between clean and non-clean edges in an image. Further, reduction of width of the Gaussian kernel improves the performance of the proposed technique as some finer details are also selected as edges. The performance of the proposed technique is expected to be improved if one uses syndrome coding to reduce the amount of distortion that occurred due to embedding.

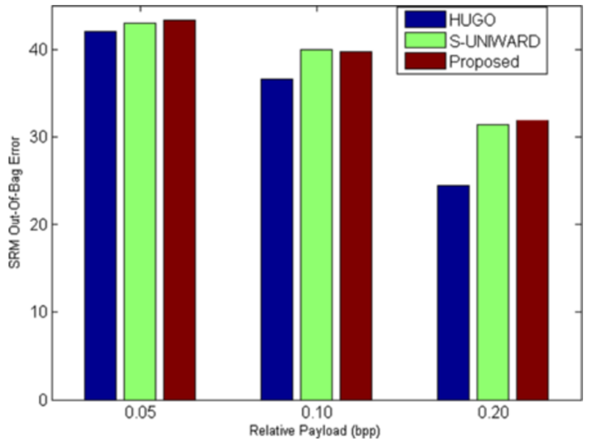

**Figure 10** SRM results for BOSSbase ver. 1.01.

## Endnotes

[a] BOWS-2 in http://bows2.ec-lille.fr/ (2013)

[b] Digital invisible ink toolkit in http://diit.sourceforge.net/files/HidingBehindCorners.pdf (2014)

**Competing interests**

The authors declare that they have no competing interests.

**Acknowledgements**

The authors would like to thank antonymous reviewers for their valuable comments and sharing their knowledge in improving this article.

Received: 3 September 2013 Accepted: 25 March 2014
Published: 27 April 2014

## References

1. P Bas, T Filler, T Pevný, ed. by T Filler, T Pevný, S Craver, and A Ker, "Break our steganographic system": The ins and outs of organizing BOSS, in *Lecture Notes in Computer Science: Information Hiding*, vol. 6958 (Springer Berlin, 2011), pp. 59–70
2. C-K Chan, L-M Cheng, Hiding data in images by simple LSB substitution. Pattern Recognit. **37**(3), 469–474 (2004)
3. A Ker, ed. by J Fridrich, Improved detection of LSB steganography in grayscale images, in *Lecture Notes in Computer Science: 6th International Workshop on Information Hiding, Toronto, Canada*, vol. 3200 (Springer Berlin, 2004), pp. 97–115
4. AD Ker, Steganalysis of LSB matching in grayscale images. IEEE Signal Process. Lett. **12**(6), 441–444 (2005)
5. T Pevnỳ, T Filler, P Bas, ed. by R Böhme, PWL Fong, and Safavi-Naini R, Using high-dimensional image models to perform highly undetectable steganography, in *Lecture Notes in Computer Science: 12th International Conference on Information Hiding, Calgary, AB, Canada* (Springer Berlin, 2010), pp. 161–177
6. AD Ker, Steganalysis of embedding in two least-significant bits. IEEE Trans. Inf. Forensics Security **2**(1), 46–54 (2007)
7. JJ Fridrich, T Pevný, J Kodovský, Statistically undetectable jpeg steganography: dead end challenges, and opportunities. Paper presented at the 9th workshop on multimedia & security (MM&Sec), Dallas, TX, USA, 20–21 Sep 2007, pp. 3–14
8. AD Ker, ed. by M Barni, J Herrera-Joancomartí, S Katzenbeisser, and F Pérez-González, A general framework for the structural steganalysis of LSB replacement, in *Lecture Notes in Computer Science: 7th International Workshop on Information Hiding, Barcelona*, vol. 3727 (Springer Berlin, 2005), pp. 296–311
9. N Provos, P Honeyman, Hide and seek: an introduction to steganography. IEEE Secur. Privacy **1**(3), 32–44 (2003)
10. A Westfeld, A Pfitzmann, ed. by A Pfitzmann, Attacks on steganographic systems - breaking the steganographic utilities EzStego, Jsteg, Steganos, and S-Tools - and some lessons learned, in *Lecture Notes in Computer Science: 3rd International Workshop on Information Hiding, Dresden, Germany*, vol. 1768 (Springer Berlin, 1999), pp. 61–76
11. S Dumitrescu, X Wu, Z Wang, ed. by FAP Petitcolas, Detection of LSB steganography via sample pair analysis, in *Lecture Notes in Computer Science: 5th International Workshop on Information Hiding, Noordwijkerhout, The Netherlands*, vol. 2578 (Springer Berlin, 2002), pp. 355–372
12. JJ Fridrich, M Goljan, D Hogea, D Soukal, Quantitative steganalysis of digital images: estimating the secret message length. Multimedia Syst. **9**(3), 288–302 (2003)
13. JJ Fridrich, M Goljan, On estimation of secret message length in LSB steganography in spatial domain. Paper presented at the SPIE security, steganography, and watermarking of multimedia contents VI, San Jose, CA, USA, 18–22 Jan 2004, vol. 5306, pp. 23–34
14. T Pevný, P Bas, JJ Fridrich, Steganalysis by subtractive pixel adjacency matrix. IEEE Trans. Inf. Forensics Security **5**(2), 215–224 (2010)
15. JJ Fridrich, J Kodovský, Rich models for steganalysis of digital images. IEEE Trans. Inf. Forensics Security **7**(3), 868–882 (2012)
16. J Kodovský, JJ Fridrich, V Holub, Ensemble classifiers for steganalysis of digital media. IEEE Trans. Inf. Forensics Security **7**(2), 432–444 (2012)
17. S Kouider, M Chaumont, W Puech, Adaptive steganography by oracle (ASO). Paper presented at the IEEE international conference on multimedia and expo (ICME), San Jose, CA, USA, 15–19 July 2013, pp. 1–6
18. V Holub, J Fridrich, T Denemark, Universal distortion function for steganography in an arbitrary domain. EURASIP J. Inform. Secur. **2014**(1) (2014). doi:10.1186/1687-417X-2014-1
19. T Filler, JJ Fridrich, Design of adaptive steganographic schemes for digital images. Paper presented at the media watermarking, security, and forensics XIII, part of IS&T SPIE electronic imaging symposium, San Francisco, CA, USA, 24 Jan 2011, vol. 7880, pp. 1–14
20. JJ Fridrich, J Kodovský, Multivariate Gaussian model for designing additive distortion for steganography. Paper presented at the IEEE international conference on acoustics, speech and signal processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013, pp. 2949–2953
21. D-C Wu, W-H Tsai, A steganographic method for images by pixel-value differencing. Pattern Recogn. Lett. **24**(9-10), 1613–1626 (2003)
22. X Zhang, S Wang, Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security. Pattern Recogn. Lett. **25**(3), 331–339 (2004)
23. C-H Yang, C-Y Weng, S-J Wang, H-M Sun, Adaptive data hiding in edge areas of images with spatial LSB domain systems. IEEE Trans. Inf. Forensics Security **3**(3), 488–497 (2008)
24. W Luo, F Huang, J Huang, Edge adaptive image steganography based on LSB matching revisited. IEEE Trans. Inf. Forensics Security **5**(2), 201–214 (2010)
25. K Hempstalk, Hiding behind corners: using edges in images for better steganography. Paper presented at the second computing women congress (CWC), Hamilton, New Zealand, 11–19 Feb 2006, pp. 1–4
26. S Dumitrescu, X Wu, N Memon, On steganalysis of random LSB embedding in continuous-tone images. Paper presented at the international conference on image processing, Rochester, NY, USA, 22–25 Sept 2002, vol. 3, pp. 641–644
27. AD Ker, R Böhme, Revisiting weighted stego-image steganalysis. Paper presented at the SPIE electronic imaging, security, forensics, steganography, and watermarking of multimedia xontents X, Orlando, FL, USA, 27–31 Jan 2008, vol. 6819, pp. 5–1517
28. S Tan, B Li, Targeted steganalysis of edge adaptive image steganography based on LSB matching revisited using b-spline fitting. IEEE Signal Process. Lett. **19**(6), 336–339 (2012)
29. RO Duda, PE Hart, DG Stork, *Pattern Classification*, 2nd edn. (Wiley, New York, NY, 2001)
30. G Gul, F Kurugollu, ed. by T Filler, T Pevný, S Craver, and A Ker, A new methodology in steganalysis: breaking highly undetectable steganography (HUGO), in *Lecture Notes in Computer Science: 13th International Conference on Information Hiding, Prague, Czech Republic*, vol. 6958 (Springer Berlin, 2011), pp. 71–84
31. V Holub, J Fridrich, Digital image steganography using universal distortion. Paper presented at the first ACM workshop on information hiding and multimedia security, Montpellier, France, 17–19 June 2013, pp. 59–68
32. V Holub, J Fridrich, Designing steganographic distortion using directional filters. Paper presented at the IEEE international workshop on information forensics and security (WIFS), Tenerife, Spain, 2–5 Dec 2012, pp. 234–239
33. T Filler, J Judas, J Fridrich, Minimizing additive distortion in steganography using syndrome-trellis codes. IEEE Trans. Inf. Forensics Security **6**(3), 920–935 (2011)
34. S Islam, P Gupta, Revisiting least two significant bits steganography. Paper presented a the 8th international conference on intelligent information processing (ICIIP), Seoul, Republic of Korea, 1–3 April 2013, pp. 90–93
35. MR Modi, S Islam, P Gupta, ed. by D-S Huang, V Bevilacqua, JC Figueroa, and P Premaratne, Edge based steganography on colored images, in *Lecture Notes in Computer Science: 9th International Conference on Intelligent Computing (ICIC)*, vol. 7995 (Springer Berlin, 2013), pp. 593–600
36. J Canny, A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
37. C Cortes, V Vapnik, Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
38. T Pevný, JJ Fridrich, AD Ker, From blind to quantitative steganalysis. IEEE Trans. Inf. Forensics Security **7**(2), 445–454 (2012)

doi:10.1186/1687-417X-2014-8
**Cite this article as:** Islam *et al.*: **Edge-based image steganography.** *EURASIP Journal on Information Security* 2014 **2014**:8.