A secure image steganography based on Hamming codes and image block complexity estimation using a zig-zag order

DOI:https://doi.org/10.54654/isj.v2i25.1123

Nguyen Duc Tuan

Abstract— **Data hiding in digital images has received considerable attention in recent years. Research efforts have primarily focused on increasing embedding capacity while preserving the visual quality of stego-images. In this paper, we propose a data hiding scheme based on Hamming codes. To enhance visual quality, the scheme estimates block complexity from pairs of adjacent pixels arranged in zig-zag order and uses this measure to identify high-texture regions for embedding message bits. To further minimize distortion, secret bits are embedded using the proposed Hamming code-based method. Moreover, embedding capacity is increased by utilizing multiple pixel bit-planes. A Canonical Gray Code (CGC) is employed in the bit-plane decomposition process to improve the accuracy of texture characterization in data hiding. Experimental results demonstrate that the proposed scheme achieves higher embedding capacity, improved visual quality, and stronger resistance to detection attacks.**

Tóm tắt— **Giấu tin sử dụng ảnh số đang nhận được nhiều sự quan tâm trong những năm gần đây. Các nhà nghiên cứu đã tập trung vào việc nâng cao dung lượng tin có thể giấu được trong khi vẫn giữ được chất lượng hình ảnh của ảnh ẩn. Trong bài báo này, một lược đồ giấu tin dựa trên mã Hamming được giới thiệu. Để nâng cao chất lượng cảm quan của ảnh mang tin, độ phức tạp của khối điểm ảnh, được xác định dựa trên các cặp điểm ảnh liền kề theo trật tự dích dắc được sử dụng để lựa chọn các khối điểm ảnh sử dụng để giấu tin. Để giảm thiểu hơn nữa suy biến xảy ra với ảnh mang tin, các bit tin mật được nhúng bởi phương thức đề xuất dựa trên các mã Hamming. Hơn nữa, dung lượng tin giấu được tăng lên nhờ kỹ thuật sử dụng nhiều hơn một lớp bit của điểm ảnh để nhúng tin. Mã gray phản xạ Canonical Gray Code (CGC) cũng được sử dụng để phân tách các lớp bit của điểm ảnh nhằm nâng cao độ chính xác trong việc xác định độ phức tạp của khối điểm ảnh khi nhúng tin. Các kết quả thực nghiệm cho thấy lược đồ đề xuất mang lại dung lượng tin giấu được cao hơn, chất lượng cảm quan của ảnh mang tin được cải thiện và bền vững hơn trước các dạng tấn công phát hiện.**

Keywords*— *Hamming codes; Image Steganography; Block Complexity Estimation; Zig-zag Order; Universal Steganalysis; Bit-plane decomposition; Security; Canonical Gray Code.

Từ khóa*— *Các mã Hamming; Nhúng tin vào ảnh số; Ước lượng độ phức tạp của khối; Trật tự dích dắc; Phân tích lớp bit; An ninh; Mã gray phản xạ.

I. INTRODUCTION

Today, to protect valuable data transmitted via the Internet, a sender may encrypt it before transmission. However, encrypted data can inadvertently signal to potential attackers the importance of the information. In contrast, steganography conceals valuable data within a cover medium in a way that raises no suspicion, making it appear as ordinary content. In general, data hiding techniques exploit the redundancy in digital media to embed information. Among these, digital images are the most common carriers due to their ubiquity on the Internet and the large amount of redundant information that can be modified with minimal impact on the human visual system (HVS) [1].

In data hiding, three key requirements are considered: visual quality, embedding capacity, and resistance to extraction attacks [2]. Although

This manuscript was received on June 28, 2025. It was reviewed on August 25, 2025, revised on September 01, 2025 and accepted on September 03, 2025.

these factors may be traded off depending on user objectives, researchers consistently aim to design algorithms that maximize embedding capacity and security while maintaining high visual quality. Adaptive image steganography techniques exploit the reduced sensitivity of the human eye to subtle changes in complex regions and use the sharpness of image areas to estimate potential embedding capacity.

To minimize visible distortions in stego images, researchers have developed embedding algorithms that incorporate edge detection methods [3 - 5]. The LSB replacement method is subsequently employed to hide the secret message within the complex image regions identified using edge detection techniques [6 - 10]. For the purpose of further decreasing the distortion caused by embedding, several researchers proposed the data hiding methods that employ Hamming codes [11 - 13]. Lee et al.[14] introduced an adaptive high-fidelity steganographic scheme, in which the edge detection is applied to identify the complex regions in a cover image. The hybrid hamming codes are utilized to conceal the given message bits into image regions identified by edge detection. As a result, it minimizes distortion in the stego image while enhancing security against detection methods. Nevertheless, this approach chooses pixel blocks over individual binary pixels on the bit-plane, thereby reducing flexibility. To further minimize the embedding degradation in comparison with the methods based on LSB replacement, Luo et al. [15] proposed the approach that utilize a LSB matching revisited method as embedding technique (EA_LSBMR). In this approach, the image regions are adaptive selected according to the size of the given message. Unfortunately, the maximum available embedding capacity is limited. Moreover, similar to Lee's method, the method EA_LSBMR also does not individually consider bit-planes of the selected image regions.

To achieve the high embedding capacity while maintaining and even enhancing the imperceptibility aspect, a lot of image steganography approaches [16 - 18] has been developed based on Bit-plane complexity segmentation (BPCS). In these methods, a cover image is initially decomposed into a set of *n* binary images before being used to conceal the message bits. These message bits are embedded within the image regions, which are identified as complex regions using the value of the summation of the number of color-changes along the rows and columns in an image [16]. Regrettably, employing this principle to gauge the complexity of image regions may result in lower accuracy [17].

To overcome the limitations of the existing data hiding methods, we focus on proposing a novel image steganography scheme in this paper. The major contributions of this paper are summarized as follows.

- A complexity estimation method based on zig-zag scanning is developed to identify textured regions for data hiding by evaluating pixel variations in horizontal, vertical, and diagonal directions.
- A new multiple bit-plane data hiding algorithm is proposed to increase embedding capacity.
- A new binary block selection method is proposed to optimally select high-textured blocks in the bit-planes of cover pixel blocks, providing flexibility in controlling embedding capacity and visual quality.
- To minimize degradation in perceptual quality, (5, 3) and (7, 4) Hamming codes have been successfully applied in developing a new image steganography scheme.
- A Canonical Gray Code (CGC) is employed to enhance precision in the complexity measurement of binary blocks on the bit-planes of pixels, thereby improving the accuracy of block selection.

The remaining of this paper is organized as follows. The literature review has been done and presented in Section 2. In Section 3, the proposed scheme is introduced with the detailed descriptions of the embedding and extraction stages. Section 4 provides the experiment details and results of the proposed scheme. Finally, the conclusion is given in Section 5.

II. Related work

In accordance with the principles of the proposed scheme, the pixel block

complexity-based steganographic approach and the Hamming code-based steganography approach are described in detail in this section. The advantages and limitations of each previous approach are discussed, along with examples of how they can be incorporated into the proposed scheme.

A. Steganography methods employing complex image regions

In general, the HVS is less sensitive to the change in the complex region in a digital image. Hence, several image steganography techniques were developed to exploit this characteristic to archive the high capacity and imperceptibility. Bit-plane complexity segmentation (BPCS) is the common approach, in which the textured image regions are used to carry the given message bits. In BPCS techniques [16-18], the embedding payload could be improved by utilizing more bit-planes of the cover images to conceal given data.

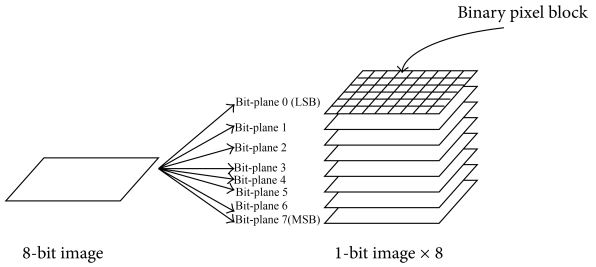


Figure 1. The bit-planes of an 8-bit graysale image

A binary image region (a block of bits on a bit-plane) is judged as complex based on the value of α:

$$\alpha = \frac{k}{2 \times m \times (m-1)}, \tag{1}$$

where *k* is the total length of the boundary in the considered block, *m* is the row or column of the block, and α is between 0 and 1. The α is compared with a predefined value (referred to as a threshold) to determine whether the binary image block is complex or not. Subsequently, the data can be embedded within these high-textured binary image blocks by utilizing any appropriate data hiding techniques. Nevertheless, this complexity measurement takes account of the pixel's value changing vertical and horizontal of the image blocks. This principle leads to confusion when determining whether a pixel block is complex or not.

To individually assess the complexity of each individual pixel in an image, Sabeti et al. [19] consider 8 neighboring pixels of a pixel to compute a complexity of it. The pixel's complexity is measured as following equation:

$$comp(i,j) = \sum_{u=-1}^{1} \sum_{v=-1}^{1} \left| cI(i,j) - cI(i+u, j+v) \right|, \tag{2}$$

where cI is a cover image. It can be observed from Eq. 2, the complexity of the pixel is the sum of absolute values of differences of the considered pixel and its 8-neighbors. Based on the definition of the complexity, the higher value of $comp(i,j)$ indicates that the pixel (i,j) is located in a high texture image region. Unfortunately, according to the Eq. 2, if the pixel (i,j) is zero and all of its neighbors are one, then the estimated complexity of this pixel is very high even if it belongs to the smooth region.

B. Hamming Code-Based Steganography Approaches

Hamming codes are a type of linear error-correcting code that can be utilized in data hiding to reduce the distortion to the stego-images [20]. To further improve the imperceptibility of the hidden message, a modified matrix embedding (MME), in which we could select the best embedding solution to be performed, was introduced by Kim et al [21].

In the MME method, a pair of bit positions (β, γ) should be flipped instead of the corresponding position α, which is identified by ($\beta \oplus \gamma$). For any α, there exist $\frac{n-1}{2}$ pairs $(\beta_1, \gamma_1), (\beta_2, \gamma_2), \ldots, (\beta_{\frac{n-1}{2}}, \gamma_{\frac{n-1}{2}})$ that satisfy the condition $\beta \oplus \gamma = \alpha$ [21]. Consequently, the MME approach has been employed in several steganographic methods, such as those proposed in [21, 22]. In these techniques, the available embedding solutions are first enumerated, and the embedding distortion corresponding to each solution is then calculated. The modification solution associated with the minimum embedding error is selected for implementation [22].

To provide a clear understanding of the principle of how MME is applied to image

steganography, a detailed explanation is given as follows. Given $\alpha = 3$, the pairs (β, γ) are enumerated as $(4,7),(1,2),(5,6)$. Assume that the detectable distortion is defined by the following equation:

$$E_{i,j} = \frac{1}{G_{i,j}}, \tag{3}$$

where $E_{i,j}$ and $G_{i,j}$ denote the detectable degradation and the gradient value of pixel (i,j), respectively. Applying this equation to the pairs (β, γ), we obtain

$$E = E_{\beta,i} + E_{\gamma,j}, \tag{4}$$

which represents the total estimated distortion of the pair. The pair of pixels corresponding to the lowest E value is then selected for modification in order to embed the information bits [23].

III. Proposed scheme

This section explains a threshold complexity estimation algorithm, as well as the embedding and extraction processes of the proposed scheme. The MME method is utilized to develop an innovative data hiding algorithm (named as HamZigZag) for embedding secret bits in textured image regions. The utilized image areas are identified based on sectioning using a zig-zag scanning order, as illustrated in Figure 2. As presented in this figure, the proposed scheme begins by performing bit-plane slicing on a cover image C. The output F, which consists of 8 bit-planes, is used to embed the given message bits using the HamZigZag method.

The proposed scheme supports bit-plane decomposition using either Pure Binary Code (PBC) or Canonical Gray Code (CGC). A value of $bCGC = 1$ (as shown in Figure 2) indicates that CGC is employed in bit-plane slicing process. In the proposed scheme, the number of bit-planes used to embed data is determined by ThrCompEst method. The pixel blocks in the specific bit-plane (to conduct a binary image) are identified based on the determined complexity threshold.

Then these selected pixel blocks are employed in data hiding. Moreover, to increase the EC, more than one bit-plane of pixels is utilized to embed the given message bits. The number of used bit-planes is determined based on the length of the input data.

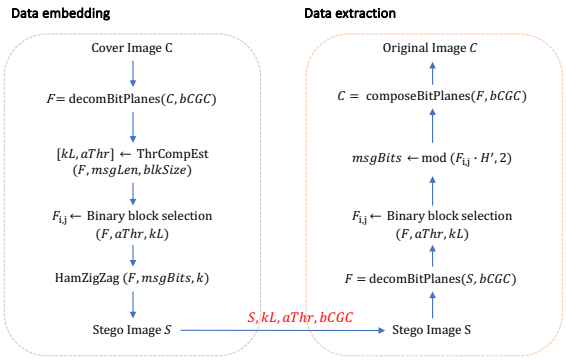
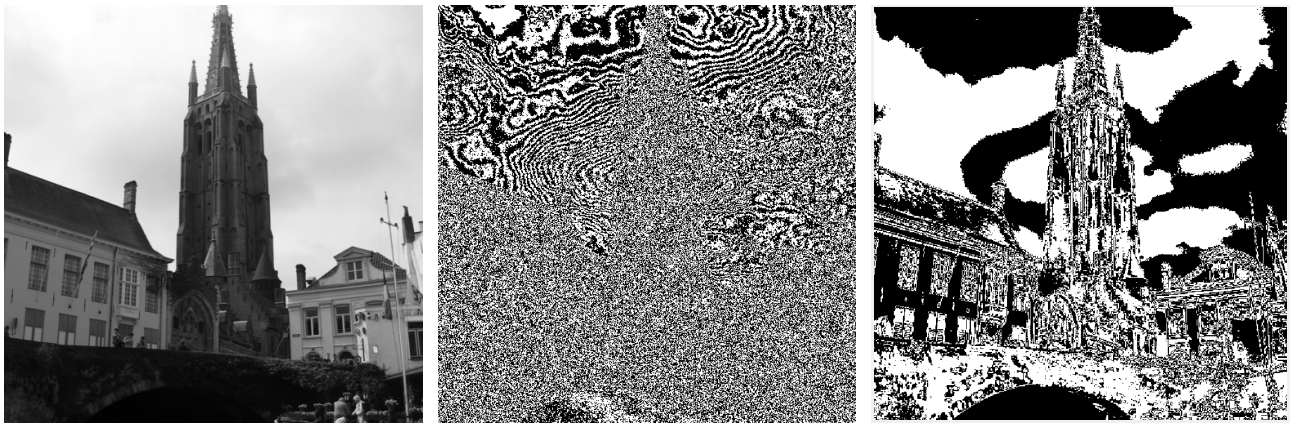


Figure 2. A flow diagram of the proposed Scheme

A. An Algorithm for Complexity Threshold Estimation Based on Zig-Zag Ordering

In general, embedding secret message bits in complex image regions reduces the risk of detection by both universal and specific steganalysis methods. The proposed algorithm is developed to determine an optimized complexity threshold for completely embedding the given message bits. To assess the complexity of image regions, Eq. 5 is applied to pixels selected in a zig-zag ordering scan.

The proposed complexity threshold estimation algorithm consists of two major phases. The first stage involves estimating the number of binary image blocks whose complexity ranges from the highest value down to 1 for each bit-plane of the cover image. In general, the highest complexity threshold value is $mB \times nB$ (where mB and nB are the number of columns and rows of the binary image block, respectively) when all adjacent pixel pairs differ. The number of binary image blocks and the supported information are then stored in an array *sol*, as presented in lines 5–7 of the algorithm 1. This is a two-dimensional array of size 245×3. The first column lists the complexity values, the second column indicates the corresponding bit-plane, and the last column records the estimated number of binary image blocks in each respective bit-plane. Based on Figure 3, it is clear that the sky regions and the wall of the house consist of several smooth areas. Grayscale images can be decomposed into 8-bit planes (as illustrated in Figure 1), ranging from the Least Significant Bit (LSB) at bit 0 to the Most Significant Bit (MSB) at bit 7 [24]. When using



(A) (B) (C)

Figure 3. (A) Cover image 22 (Bows-2), (B) Corresponding LSB bit-plane, and (C) Corresponding 5th bit-plane

Data: $C, msgLen, imgW, imgH, blkSize$
Result: $kL, aThr$
/* Maximum number of bit-planes should be used to carry message bits */
$F = decomBitPlanes(C, CGC)$;
$iNumOfBitplanes \leftarrow 5$;
$ThrInit = [\ 0 \quad 14 \quad 14 \quad 14 \quad 6\ ]$;
$aThr(1:5) \leftarrow 0$;
$i\_start \leftarrow 0$; $i\_fin \leftarrow 0$;$j\_start \leftarrow 0$;$j\_fin \leftarrow 0$;$sol(:,3) \leftarrow 0$;
/* sol is a 2D array of size 254×3 */
/* Estimate the number of available image blocks for each threshold. */
for *each bit-plane i from 2 to 5* **do**
  $sol(i,3) \leftarrow NumOfImgBlock$ /* Number of image blocks that match the selection complexity threshold */
end
$AvailBlkcOfEachLayer(1:iNumOfBitplanes) \leftarrow 0$;
while $(true)$ **do**
  if $(sum(AvailBlkcOfEachLayer) * (blkSize * 3) < msgLen \&\& sol(idxA(i,5),1) >= ThrInit(5))$ **then**
    $i = i + 1$;
    Add $sol(idxA(i,5),3)$ to $AvailBlkcOfEachLayer(5)$;
  else
    $aThr[5] = sol(idxA(i,5),1)$;$kL = 1$; break;
  end
end
/* Bit planes 2 to 4 are used to further embed message bits. */
if $(AvailBlkcOfEachLayer(5) * (sizeOfBlock * 3) < lendata)$ **then**
  while $(true)$ **do**
    if $(sum(AvailBlkcOfEachLayer) * blkSize * 3 < msgLen)$ **then**
      Add $sol(idxA(i,iK),3)$ to $AvailBlkcOfEachLayer(iK)$;
      if $(i < numberOfElementsInSol \&\& sol(idxA(i,iK),1) >= ThrInit(iK))$ **then**
        $i = i + 1$;
      else
        $aThr[iK] \leftarrow sol(idxA(i,iK),1)$;
        if $(sum(AvailBlkcOfEachLayer(:) * (sizeOfBlock * 3)) < msgLen \&\& iK < 4)$ **then**
          $iK = iK + 1; i = 1$;
        else
          $break$;
        end
      end
    else
      $aThr[5] \leftarrow sol(idxA(i,5),1)$;
    end
  end
else
  $aThr[iK] \leftarrow sol(idxA(i,iK),1)$;
  $kL \leftarrow sum(aThr > 0)$;
end

Algorithm 1. ThrCompEst: A complexity threshold estimation algorithm using zig-zag scanning order

the LSB bit-plane (bit-plane 0th, as shown in Figure 3A) to assess the complexity of these areas,

the LSB bit-plane (bit-plane 0th, as shown in Figure 3a) to assess the complexity of these areas, they may be erroneously categorized as complex regions within the LSB bit-plane. This is clearly illustrated in Figure 3b, where it is evident that sky regions can be incorrectly identified as complex image regions.

To increase the maximum number of message bits that can be embedded into a cover image while preserving an appropriate visual quality, the proposed scheme employs four bit-planes (0–3) of the cover pixels to carry the message bits, as shown in lines 14 - 29 of the algorithm. To enhance the imperceptibility of the hidden data against bit-plane-based steganalysis, flat image regions are excluded from data hiding. In the proposed scheme, the 5th bit-plane is used to identify binary image regions on the LSB bit-plane, since it provides better texture characteristics than the LSB bit-plane in the bit-plane slicing process as shown in Figure 3.

The second stage involves determining the complexity threshold for each bit-plane to ensure that the given message bits are completely embedded in the cover image. To prevent smooth image regions from being used for data hiding at high embedding rates, the lowest complexity thresholds are initially set to *ThrInit*, defined as $[0, 14, 14, 14, 6]$. In other words, *ThrInit* acts as a lower bound, ensuring that the estimated thresholds do not fall below it.

These values are determined based on the following principle. In the proposed scheme, to better capture the complexity of pixel blocks (image regions) in the 5th bit-plane of the carrier image, the complexity threshold of the 5th bit-plane is used instead of that of the LSB bit-plane when identifying embedding regions. Consequently, the initial complexity threshold of the LSB bit-plane is set to 0.

The initial values of the complexity threshold for the next three bit-planes are set to 14, as this value helps prevent significant alterations to pixel blocks according to experimental diagnostics.

If the number of binary image blocks that satisfy the complexity threshold in the LSB bit-plane is insufficient to embed the given message bits, the threshold estimation process proceeds to higher bit-planes (from the 2nd to the 4th bit-plane). For each bit-plane, the complexity threshold is incremented by one, and the number of eligible binary image blocks is recalculated until it reaches the predefined value. This procedure is described in lines 19 - 21 of the proposed algorithm 1.

This algorithm assesses the complexity of a binary image block by calculating the cumulative difference between pairs of adjacent pixels in a zig-zag scanning order. This approach considers the pixel's value changing in three different directions, resulting in improved accuracy in determining the complexity characteristics of pixel blocks.

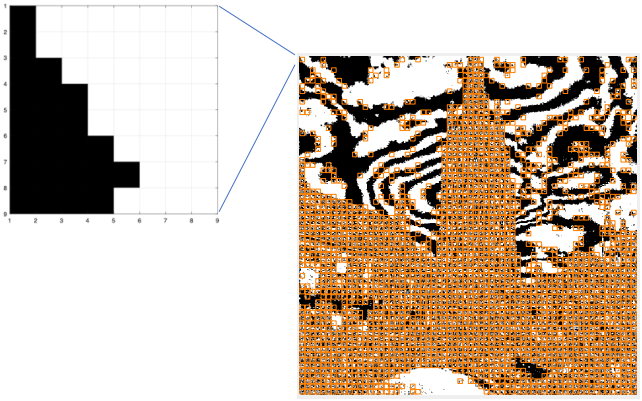


Figure 4. An 8 x 8 binary image block extracted from the 2nd bit-plane of image 22.bmp in the Bows-2 database

To illustrate the accuracy of the complexity estimation process of the proposed scheme, the complexity of the binary image block shown in Fig. 4 was estimated using both run-length irregularity β [17] and the proposed approach based on the zig-zag scanning order. The binary image corresponds to the 2nd bit-plane of image 22.bmp. The estimated values were 0.3972 for the run-length irregularity and 0.2245 for the proposed scheme, respectively. These results suggest that the complexity estimation based on the zig-zag scanning order provides more reliable results than the run-length irregularity-based technique. This observation is supported by the fact that the binary pixel block in Figure 4 is classified as complex according to the run-length irregularity method.

Figure 5 illustrates that the use of the run-length irregularity-based technique for measuring texture characteristics resulted in the erroneous selection of numerous low-complexity pixel blocks (Figure 5a). In contrast, the zig-zag scanning order-based estimation method achieved higher accuracy in determining texture characteristics (Figure 5b).

Data: A cover binary image C, $msgBits$, $aThr$
Result: A stego binary image S
$msgLen \leftarrow len(msgBits)$;
$F = decomBitPlanes(C, bCGC)$;
$iNumOfPixBlk = \lfloor img\_size/blkSize \rfloor$;
while $(true)$ **do**
  $i\_start \leftarrow (i * blkSize) + 1$; $i\_fin = (i * blkSize) + blKSize$;
  $j\_start \leftarrow (j * blkSize) + 1$; $j\_fin = (j * blkSize) + blKSize$;
  $[comp] = estiCompZigZag(F(i\_start : i\_fin, j\_start : j\_fin, k))$;
  if $(k == 1)$ **then**
    if $(comp >= aThr[5])$ **then**
      $msgBitBlk = msgBits(pixBlockIndx * MsgBitsPerBlk + 1 :$
       $pixBlockIndx * MsgBitsPerBlk + MsgBitsPerBlk)$;
      $F[i\_start : i\_fin, j\_start : j\_fin, k] = HamZigZag(F[i\_start : i\_fin, j\_start : j\_fin], msgBitBlk, k)$;
      $NumOfUsedBlkc = NumOfUsedBlkc + 1$;
      $pixBlockIndx = pixBlockIndx + 1$;
    end
  else
    if $(comp >= aThr[k])$ **then**
      $msgBitBlk = msgBits(pixBlockIndx * MsgBitsPerBlk + 1 :$
       $pixBlockIndx * MsgBitsPerBlk + MsgBitsPerBlk)$;
      $F[i\_start : i\_fin, j\_start : j\_fin, k] = HamZigZag(F[i\_start : i\_fin, j\_start : j\_fin], msgBitBlk, k, comp)$;
      $NThr = estiCompZigZag(F[i\_start : i\_fin, j\_start : j\_fin, k])$;
      if $(NThr >= aThr[k])$ **then**
        $NumOfUsedBlkc = NumOfUsedBlkc + 1$;
        $pixBlockIndx = pixBlockIndx + 1$;
      end
    end
  end
  $MsgBitsEmbedded = NumOfUsedBlkc * MsgBitsPerBlk$;
  if $(j < (iNumOfPixBlk - 1)\&\&(MsgBitsEmbedded < msgLen))$ **then**
    $j = j + 1$;
  else
    if $(i < (iNumOfPixBlk - 1)\&\&(MsgBitsEmbedded < msgLen))$ **then**
      $j = 0; i = i + 1$;
    else
      if $(k < kL\&\&(MsgBitsEmbedded < msgLen))$ **then**
        $k = k + 1; j = 0; i = 0$;
      else
        break;
      end
    end
  end
end
$S = composeBitPlanes(F, bCGC)$;

Algorithm 2: A Proposed Hamming-Based Data Embedding Algorithm

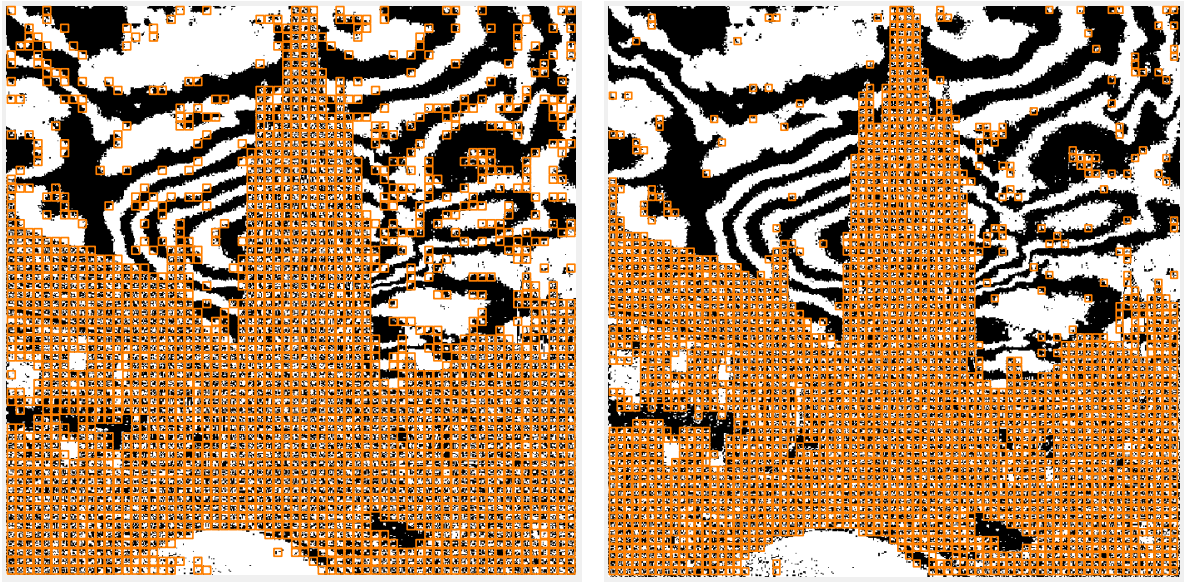
B. A proposed hamming-based data embedding algorithm

During the initial stage of the embedding process, the binary blocks in the LSB bit-plane are utilized for data hiding. These candidate blocks are selected by comparing their estimated complexity with a predefined threshold derived from bit-plane 5, as determined by the ThrCompEst algorithm. The complexity of each binary block is computed using the estiCompZigZag method. In the proposed data hiding algorithm, the 1st bit-plane (LSB) is first employed to embed the given bits. At this stage, the complexity of the corresponding binary block in bit-plane 5 is used to guide the selection of blocks in bit-plane 1. This is because the determination process achieves higher precision when the 5th bit-plane is used, as it provides a better distinction of textured regions compared to the 1st bit-plane.

If there are still remaining message bits to be embedded, the process continues with the binary image blocks in the 2nd through 4th bit-planes. The complexity values of these blocks are estimated and compared with the corresponding complexity thresholds of each bit-plane to identify those suitable for concealing the message bits. The embedding process terminates once all the given message bits have been successfully hidden within the selected pixel blocks.

1. A proposed Hamming-based data embedding method using a zig-zag scanning order

Hamming-based steganography methods typically offer the flexibility to select an optimized modification solution for embedding



(A) (B)

Figure 5. The selected image blocks on the 3rd bit-plane of image 22.bmp, determined by (A) the run-length irregularity-based method and (B) the zig-zag scanning order-based method

message bits into a cover block of bits. Consequently, in this proposed scheme, two Hamming codes, (5, 3) and (7, 4), are utilized to establish a novel data hiding method that guarantees that the complexity of the resulting stego binary image block remains unchanged or even increases, adhering to the predefined complexity threshold. This anticipated outcome can be achieved by employing the modification solution that does not result in a reduction in complexity.

Generally, the (7, 4) Hamming code provides more embedding solutions than the (5, 3) code. However, to achieve a higher embedding payload, the (5, 3) Hamming code-based data hiding method is employed. In this scenario, the lookup table presented in Table 2 of [25] is used to simplify the identification of the bit to be altered in the cover blocks.

According to the principle of the Hamming code-based data hiding method, the positions where the cover bits should be altered are identified and stored in the variable $pos2Mod$. If $pos2Mod = 0$, the given message bits are embedded into the cover bits without any modification. Otherwise, when $pos2Mod \neq 0$, at most two cover bits may be flipped according to one of the following five cases.

- Case 1. **If** $conBlk(pos(ii))$ is the first element of $conBlk$, and $conBlk(pos(ii)+1) = conBlk(pos(ii))$, **then** $conBlk(pos(ii)) \leftarrow conBlk(pos(ii)) \oplus 1$, and then set $iCase(ii) = 1$.

- Case 2. **If** $conBlk(pos(ii)) = conBlk(pos(ii)-1)$, **then** $conBlk(pos(ii)) \leftarrow conBlk(pos(ii)) \oplus 1$ and then set $iCase(ii) = 2$

- Case 3. **If** $conBlk(pos(ii))$ is the first bit of $conBlk$, and its value is equal to the last bit of the previous sub-array in $blkzigzag$, **then** $conBlk(pos(ii)) \leftarrow conBlk(pos(ii)) \oplus 1$ and set $iCase(ii) = 3$

- Case 4. **If** $conBlk(pos(ii))$ is the last bit of $conBlk$ and its value is equal to the first bit of the next sub-array in $blkzigzag$, **then** $conBlk(pos(ii)) \leftarrow conBlk(pos(ii)) \oplus 1$ and set $iCase(ii) = 4$

- Case 5. **If** $conBlk(pos(ii))$ is not the first or last bit of $conBlk$, and its value is equal to the previous and next bits in $conBlk$, **then** $conBlk(pos(ii)) \leftarrow conBlk(pos(ii)) \oplus 1$ and set $iCase(ii) = 5$

These five cases are designed to ensure that the complexity of $conBlk$ does not fall below the corresponding threshold. This principle helps to

Data: A binary image block $blkzigzag$, H, msg, thr
Result: A stego block $mBlkzigzag$
Step 1: Flatten the matrix $blkzigzag$ into a one-dimensional array following a zig-zag order;
Step 2: Segement $blkzigzag$ into m one-dimensional arrays $conBlk$;
foreach *a sub-block conBlk corresponding to index i* **do**
  Step 3: Calculate a syndrome $synd \leftarrow mod(conBlk \cdot H^T, 2)$;
  $pos2Mod \leftarrow bin2dec(synd \oplus msg)$;
  if $pos2Mod > 2$ **then**
    Retrieve positions to be modified in $conBlk$ using the lookup table $lookupTable$ and store them in pos;
    foreach *position ii in pos* **do**
      Determine the case based on position and neighboring bits;
      Flip the value of $conBlk(pos(ii))$ accordingly;
    end
    Calculate the new complexity of $conBlk$ and store it in $iNewCom$;
    if $iCase$ *is empty or one bit is altered in conBlk and* $iNewCom < thr$ **then**
      Sequentially flip each bit in $conBlk$;
      Apply operations in Step 3;
      if $iNewCom \geq thr$ **then**
        break;
      end
      else
        Store the changed values in $xSol$;
      end
    end
  end
end
Step 4: Select the solution from $xSol$ with the smallest changed value to perform;
Step 5: Store the stego *conBlk* to resulting *mBlkzigzag*.

HamZigZag: A proposed Hamming-based data embedding method using a zig-zag scanning order

Data: $S, aThr, kL, bCGC, H, blkLen$
Result: $msgExtracted$
$F = decomBitPlanes(S, CGC)$;
$msgBitsE \leftarrow []; mLen \leftarrow 3; im\_size \leftarrow size(S, 1)$;
$pixBlkIdx \leftarrow 0; bklLen \leftarrow 5; iNumOfPixBlk \leftarrow \lfloor (im\_size/blkLen - 1) \rfloor$;
$t \leftarrow 1; i \leftarrow 0; j \leftarrow 0; i\_start \leftarrow 0; i\_fin \leftarrow 0; j\_start \leftarrow 0; j\_fin \leftarrow 0$;
while $(true)$ **do**
  $i\_start = (i * blkLen) + 1$;
  $i\_fin = (i * blkLen) + blkLen$;
  $j\_start = (j * blkLen) + 1$;
  $j\_fin = (j * blkLen) + blkLen$;
  if $(t == 1)$ **then**
    $iK = 5$;
  else
    $iK = t$;
  end
  $comp \leftarrow estiCompZigZag(F[i\_start : i\_fin, j\_start : j\_fin, iK])$;
  if $(comp \geq aThr[iK])$ **then**
    $sF \leftarrow F[i\_start : i\_fin, j\_start : j\_fin, t]$;
    for $kF = 1$ *to* 7 **do**
      $msg[(kF - 1) * mLen + 1 : (kF - 1) * mLen + mLen] = mod((sF(kF, :)) * H', 2)$;
    end
    $msgBitsE[pixBlkIdx * (blkLen * 3) + 1 : pixBlkIdx * (blkLen * 3) + (blkLen * 3)] = msg$;
    $pixBlkIdx = pixBlkIdx + 1$;
  end
  if $(j < iNumOfPixBlk)$ **then**
    $j = j + 1$;
  else
    if $(i < iNumOfPixBlk)$ **then**
      $j = 0; i = i + 1$;
    else
      if $(t < kL)$ **then**
        $i = 0; j = 0; t = t + 1$;
        `/* Restart the data hiding process to embed data bits in the next bit-plane */`
      else
        $break$;
      end
    end
  end
end

Algorithm 4: A proposed hamming-based data extraction algorithm

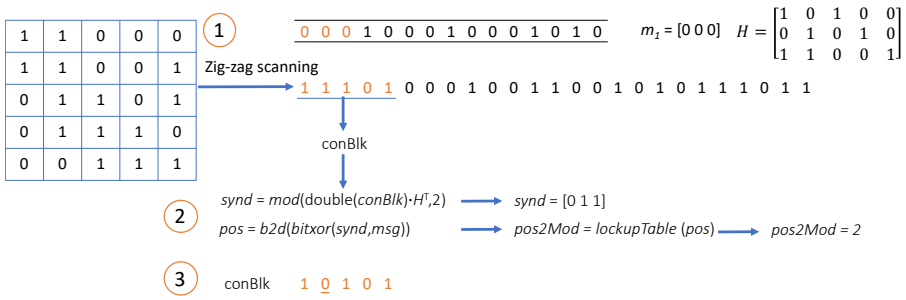


Figure 6. An example illustrating case 5 of the proposed data-hiding technique based on the (5, 3) hamming code

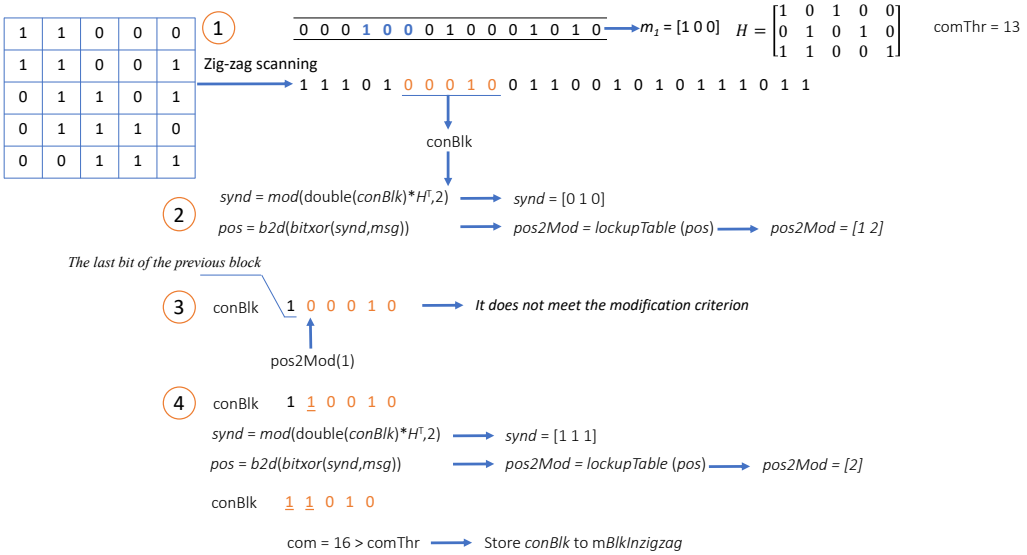


Figure 7. An example illustrates a scenario in which the most appropriate solution for modification is executed

guarantee an accuracy of data extraction. These cases involve checking the values of the binary pixels immediately preceding and following $conBlk(pos(ii))$. If these pixels are equal, the value of $conBlk(pos(ii))$ is flipped from 0 to 1 or vice versa, depending on its current value. This modification may either maintain the complexity of the binary image block or increase it beyond the threshold complexity.

If the new complexity of the stego binary image block is lower than the current threshold complexity, a cover bit in *conBlk* is sequentially flipped according to the rule in the five specified cases. The embedding process then restarts from Step 3 to identify the optimal modification solution.

To illustrate how data hiding operates in the proposed scheme, case 5 is presented in Figure 6. In this example, the b2d function is used to compute the decimal value from an array of bits. According to the principle of the data hiding algorithm, the second in the block of bits $conBlk$ is flipped from 1 to 0 to embed the message bits $[0, 0, 0]$.

Furthermore, to illustrate how the optimal embedding solutions are selected, an example is provided in Figure 7. As shown in the figure, during the second step, the identified position does not satisfy the modification condition because the preceding bit of the candidate position does not match '1'. Consequently, inverting the value at the candidate position reduces the complexity of the analyzed block by one. As a result, the candidate bit is flipped, forming a new carrier block, and the new candidate position is re-evaluated to determine the most suitable solution.

2. *A proposed hamming-based data extraction algorithm*

In the proposed scheme, the utilized parameters are used as the secret keys to protect the hidden message against the extracting attacks. To retrieve the embedded data from a stego

image, the variables $aThr$ and kL are utilized to identify the bit-planes involved in the data hiding process.

The process of message extraction begins by processing the least significant bit-plane (LSB) and continues until the kL bit-plane is reached. In the LSB bit-plane, the binary image blocks are matched with their corresponding blocks in the fifth bit-layer, i.e., bit-plane 4, using complexity characteristics. Afterwards, the concealed message bits are extracted by applying the following formula.

$$m' = mod(sF \cdot H^T, 2) \tag{5}$$

where sF represents the stego block, and T is the matrix transpose operator. A hamming matrix H contains a two dimensional array as shown in Figure 7. The detailed example presented in Figure 8 illustrates the extraction process. According to the extraction algorithm, the complexity of the considered binary pixel blocks is calculated first. Then, its result is compared with the complexity threshold used in the current bit-plane, which is stored in the variable $aThr$. If the measured complexity is larger than the threshold, the binary pixel block is segmented into 1×5 pixel blocks to extract the hidden message bits.

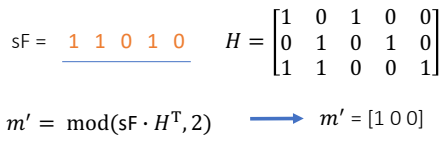


Figure 8. An example of message extraction

If the extracted message bits m' are identical to m, then the extraction process is error-free. As shown in this example, $m' = [1, 0, 0]$ is exactly the same as $m = [1, 0, 0]$, confirming the accuracy of the proposed data hiding scheme.

IV. Experiment Results and Discussion

This section presents the design, analysis and evaluation of results obtained from experiments. These obtained experimental results are compared with those from previously introduced related methods to demonstrate the superiority of the novel approach. The proposed scheme and compared approaches were implemented using Matlab 2022a on a laptop with Windows 11 installed. Experimental images were selected from two image libraries: SIPI and BOWS-2. The 10,000 gray images from Bows-2 are in PGM format and have a size of 512×512 pixels. These images were converted to the BMP file format using MATLAB's imwrite function, without applying any modifications, for use in the subsequent experiments.

A. Embedding capacity analysis

In this section, the embedding rate (ER) metric is employed to examine the embedding capacity of the proposed scheme. This metric represents the number of message bits that can be embedded in a cover image. The term ER is calculated using the following equation.

$$ER = \frac{NoMsgBits}{nW \times nH}, \tag{6}$$

where *NoMsgBits* denotes the number of message bits that can be embedded in a carrier image of size $nW \times nH$. Therefore, the unit of ER is expressed as the number of message bits per pixel (bpp). In the following experiment, SIPI images are employed to examine the maximum achievable ER of stego-images generated by the proposed scheme using two Hamming code-based methods with CGC and PBC in bit-plane decomposition. The corresponding PSNR values are presented in Table I.

It can be observed from Table I that the proposed scheme achieves a maximum embedding rate exceeding 2.0 bpp for most of the tested images. This is because the scheme exploits up to four bit-planes of the cover images to conceal the given message. However, the maximum available embedding capacity of the stego-images depends on their own texture characteristics. Therefore, in this experiment, the magnitude of used experimental data corresponds to ER of 3.0 bpp.

This experiment evaluates the effectiveness of using two Hamming codes, (5, 3) and (7, 4), for data hiding with bit-plane decomposition using CGC and PBC to minimize the reduction of texture characteristics in stego binary blocks. This reduction occurs because multiple bits per pixel are altered to carry message bits. Consequently, the number of pixel blocks with complexity below the threshold is counted and reported in Table I. In this table, the terms

TABLE I. MAXIMUM AVAILABLE EMBEDDING RATES, CORRESPONDING PSNR VALUES, AND THE NUMBER OF PIXEL BLOCKS WITH COMPLEXITY BELOW THE USED THRESHOLD FOR THE STEGO-IMAGES GENERATED BY THE PROPOSED SCHEME USING SIPI IMAGES

Image	ER (bpp)				PSNR (dB)				NoBlockBelowThR			
	Ham53 PBC	Ham53 CGC	Ham74 PBC	Ham74 CGC	Ham53 PBC	Ham53 CGC	Ham74 PBC	Ham74 CGC	Ham53 PBC	Ham53 CGC	Ham74 PBC	Ham74 CGC
Aerial	2.242	2.054	1.625	1.493	34.641	35.132	36.309	36.707	0	0	0	0
Barbara	2.161	1.861	1.564	1.339	34.842	35.763	36.545	37.355	0	0	0	0
Boat	2.319	2.105	1.672	1.528	34.392	35.007	36.040	36.579	0	0	0	0
Cameraman	1.668	1.461	1.195	1.045	35.835	36.701	37.671	38.413	0	0	0	0
Couple	2.240	1.952	1.628	1.434	34.614	35.525	36.257	37.038	0	0	0	0
F16	1.963	1.632	1.411	1.166	35.435	36.908	37.240	38.520	0	3	0	0
Lake	2.279	2.066	1.639	1.480	34.41	34.912	36.089	36.556	0	0	0	0
Lena	2.205	1.831	1.594	1.310	34.669	36.082	36.355	37.823	1	0	0	0
Mandrill	2.369	2.304	1.705	1.663	34.192	34.124	35.838	35.637	1	0	0	0
Peppers	2.293	1.928	1.654	1.374	34.377	35.573	36.045	37.353	0	0	0	0
Splash	2.062	1.611	1.499	1.153	35.093	37.366	36.837	39.236	0	0	0	0
Tiffany	2.134	1.754	1.536	1.245	34.875	36.591	36.622	38.439	0	1	0	0
Goldhill	2.305	2.096	1.661	1.526	34.363	34.856	36.023	36.394	0	1	0	0

Ham53PBC and Ham53CGC indicate that the (5, 3) Hamming code is used for data hiding, with PBC and CGC employed for bit-plane slicing. Similarly, Ham74PBC and Ham74CGC denote that the (7, 4) Hamming code is used for data hiding, with PBC and CGC employed for bit-plane decomposition of cover pixels.

As shown in Table I, the two variations of the proposed data hiding methods, Ham74PBC and Ham74CGC, produce no binary blocks with reduced texture complexity. This is because there are more possible embedding solutions available for modification to embed the secret message bits. However, the maximum available embedding payloads are smaller than those of the variations using (5, 3) Hamming codes. Moreover, although a small number of pixel blocks have complexity below the corresponding bit-plane threshold, the visual quality and embedding capacity of the stego images generated by the variations using (5, 3) Hamming codes remains unaffected.

The results in Table I show that more complex images provide greater hidden message capacity. The two images, Cameraman and F16, consist of large smooth regions, resulting in stego-images with smaller ERs compared to the other images used. For the two Hamming code-based data hiding methods, using CGC to decompose the bit-planes achieves higher embedding payloads than PBC. This indicates that the CGC technique is more effective than PBC in representing the texture features of the decomposed bit-planes. Overall, across the tested images, the (5, 3) Hamming code-based scheme achieves higher maximum embedding rates than the (7, 4) Hamming code-based scheme.

To further assess the maximum available embedding capacity of the proposed scheme, in the next experiment, a set of 2,000 cover images randomly selected from Bows-2 were embedded with an attempting payload of 3.0 bpp. The results obtained are depicted in Figure 9. As shown in this figure, according to the use of (5, 3) Hamming code for embedding the secret bits since PBC is employed for bit-plane decomposition, the stego images created by the proposed scheme consistently achieved a payload exceeding 2.0 bpp.

Figure 10 shows that the maximum embedding rates achieved by the proposed scheme with the (5, 3) Hamming code using PBC are superior to those of the Fidelity approach [14]. Similar to the proposed scheme, Fidelity also utilizes multiple bit-planes of a cover image to embed data. However, the stego images generated by Fidelity approach do not achieve embedding rates higher than 1.0 bpp. This limitation arises because Fidelity employs

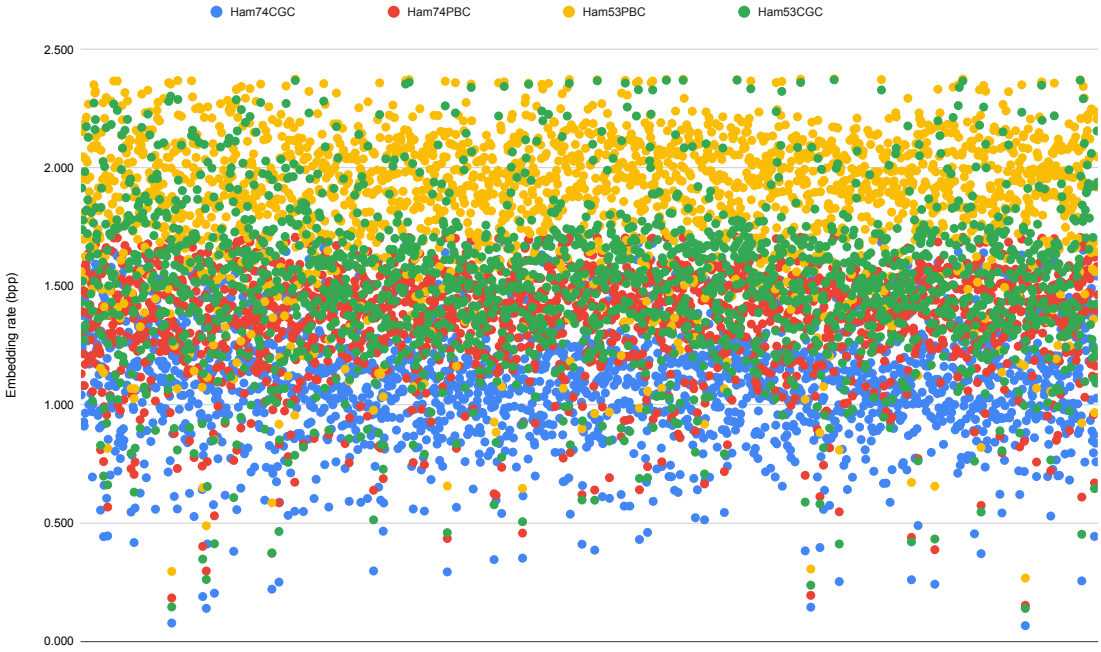


Figure 9. Distribution of the maximum embedding rates for 2,000 randomly selected Bows-2 stego images generated using the (5, 3) and (7, 4) hamming code-based methods of the proposed scheme

the Canny edge detection method to locate edge regions in the cover images, and the number of bit-planes used for embedding is determined by the sharpness degree ω. If ω equals G (where $G = 2 \times th$), the corresponding pixel block is used for embedding; otherwise, the block remains unchanged and the next block is considered. These results demonstrate that the proposed complexity-based method using zig-zag scanning is more effective than the Fidelity method.

In addition to the issue of inaccurately determining the complexity of pixel blocks, it is worth noting that the Fidelity method lacks flexibility when it comes to determining which pixel blocks will be used on different bit layers. The proposed method addresses this limitation by allowing pixel blocks on different bit layers to have varying complexities, providing greater flexibility in determining which blocks are used where. This approach overcomes one of the limitations of the Fidelity method and provides a more versatile and effective approach to image compression. The proposed scheme achieves a high embedding capacity and enhances the visual quality of stego images, compared to the Fidelity method. This advantage is verified by the experiment. As shown in Table II, the maximum payload provided by the proposed scheme is almost double that of the Fidelity method.

B. Visual Quality

The visual quality of stego images is a crucial factor in image steganography. In this experiment, the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measurement (SSIM) [28] are evaluated from cover images (from Bows-2) and their corresponding stego images. Table III presents the PSNR data of the proposed scheme with various variations, utilizing (5, 3) Hamming code and (7, 4) Hamming code, along with PBC and CGC bit-plane separation modes. The table also compares the PSNR values of images generated by previously introduced methods, including MagicCubes [29], XorEdge [7], Fidelity [14], and HyBridHam [30]. Generally, a higher PSNR value indicates a greater imperceptible quality. The HVS is unable to discern the differences between a cover and stego image due to the PSNR value exceeding 28 dB. The PSNR metric is defined as follows:

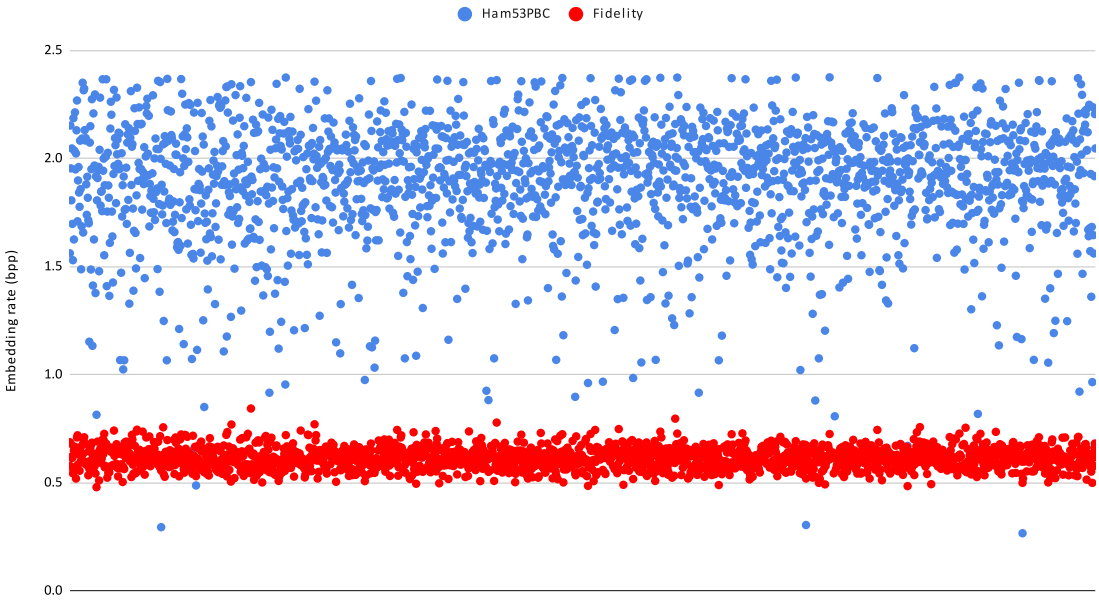


Figure 10. The maximum available embedding rates were compared for 2,000 stego images generated by the proposed approach and the fidelity method

TABLE II. MAXIMUM EMBEDDING PAYLOAD COMPARISON OF THE PROPOSED METHOD USING THE (5, 3) HAMMING CODE WITH CGC AND OTHER RELATED APPROACHES

		Lena	Lake	Baboon	Airplane	Barbara	Peppers	Boat
bpp	Ham53CGC	1.484	1.585	1.737	1.323	1.475	1.537	1.636
	AccTexCmp [26]	1.465	-	1.446	1.360	1.477	1.057	1.464
	BEASS [27]	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	Fidelity [14]	0.7425	0.7985	1.1747	0.7705	0.8663	0.7075	0.8774

$$PSNR = \frac{255^2}{MSE}$$

$$where\ MSE = \frac{1}{nW \times nH} \sum_{0}^{nW-1} \sum_{0}^{nH-1} \|I - K\| \quad (7)$$

where nW, nH are the width and height of the image, I and K are cover and stego images, respectively.

The average PSNR values presented in Table III indicate that the proposed scheme offers superior visual quality compared to the compared methods when embedding rates are within the range of [0.05 - 0.40] bpp. However, since the message payloads are 0.45 bpp and 0.50 bpp, respectively, the proposed scheme is unable to maintain high visual quality when compared to other previous methods. This is because at high embedding rates, the number of utilized pixel blocks increases to accommodate more message bits, particularly in the high bit-planes of pixels. Furthermore, the avoidance of smooth binary image regions leads to an increase in the number of pixel blocks employed in high bit-planes, resulting in a corresponding increase in visual distortion.

To further assess the visual quality of the proposed scheme, a subsequent experiment is conducted. In this experiment, the PSNR values of the cover images (obtained from the SIPI dataset) with their corresponding stego images generated by the proposed scheme and a high-quality related method, as referenced in [10], are compared. Table IV presents the PSNR values for both the existing and proposed schemes for the six utilized images. The results demonstrate that the proposed scheme exhibits superior visual quality compared to the existing

TABLE III. THE COMPARISON OF THE PROPOSED SCHEME AND THE PREVIOUS TECHNIQUES IN TERM OF VISUAL QUALITY UNDER VARIOUS OF EMBEDDING RATES

	Method							
ER (bpp)	H74PBC	H74CGC	H53PBC	H53CGC	Magic Cubes [29]	Xor Edge [7]	Fidelity [14]	HybridHam [30]
0.05	66.5041	66.4926	65.4269	65.4185	62.4684	64.6000	62.1542	63.7177
0.10	63.4612	63.4281	62.3874	62.3737	59.4582	61.9800	57.8662	60.7064
0.15	61.6540	61.4136	60.5985	60.5338	57.6966	-	59.5013	58.9425
0.20	60.2896	59.4033	59.3102	59.0316	56.4477	58.9600	55.7317	57.6909
0.25	59.0427	57.1134	58.2561	57.4817	55.4770	57.9900	56.6824	56.7198
0.30	57.5803	54.9870	57.3021	55.8669	54.6867	57.1900	54.1287	55.9264
0.35	55.7869	53.2775	56.3038	54.2381	54.0170	-	54.9057	55.2556
0.40	54.0295	51.8893	55.1443	52.8876	53.4366	56.1200	52.6537	54.6751
0.45	52.6040	50.7566	53.8823	51.7365	52.9255	-	53.3669	54.1623
0.50	51.5170	49.8230	52.6433	50.7577	52.4678	-	52.0150	53.7036

method under various embedding rates. This outperformance of the proposed scheme is attributed to the reduced number of bit modifying performed by Hamming codes-based data hiding method to embed the same message bits in comparison with the related methods. Specifically, if the position of bits needed to alter is equal to zero, no modifying is applied to the consider binary blocks. Consequently, the degradation to stego-images is minimized, and the visual quality is preserved.

Analogously to prevalent image steganography techniques, embedding data bits not only results in perceptible visual distortion but also structural distortion. Consequently, the subsequent experiment assesses structural distortion employing the Structural Similarity Index Measure (SSIM). This metric quantifies the similarity between the carrier and stego images. An SSIM value of 1.0 signifies that the message is seamlessly embedded within the stego image without any structural distortion. The SSIM value is computed as follows:

$$SSIM(x,y) = [L(x,y)]^{\alpha} \cdot [C(x,y)]^{\delta} \cdot [S(x,y)]^{\gamma} \quad (8)$$

where L, C, and S represent luminance, contrast, and structure, respectively. Their corresponding exponents are specified as a three-element vector of non-negative numbers in the format $[\alpha, \delta, \gamma]$.

As indicated in Table V, the proposed scheme's variants resulted in fewer structural modifications in the stego images compared to previous methods for payloads less than 0.4 bpp. This outcome can be attributed to the accuracy of the complexity estimation process. However, since more mystery bits are embedded and smooth regions are avoided for data hiding, the structural degradation has increased. Consequently, the SSIM values of the stego images generated by the proposed scheme are lower than those of the previous embedding methods (MagicCubes [29], Fidelity [14], HybridHam [30]).

C. Security analysis of the Proposed Scheme

Security is an important factor in evaluating the performance of a steganography approach. It refers to the likelihood that the hidden message can be detected by specific or universal steganalysis methods. In this experiment, two steganalysis techniques, LSB Enhancement, the Ensemble Classifier, and XuNet steganalysis, are employed to evaluate the security performance of the proposed scheme and compare it with previous methods.

1. Resistance to LSB Enhancement Attack

In general, the LSB bit-plane is usually affected in multiple bit-plane steganographic techniques, including the proposed scheme and the previous method implemented in this experiment. Therefore, LSB enhancement

TABLE IV. VISUAL QUALITY COMPARISON BETWEEN THE PROPOSED SCHEME AND DIAHYED METHOD [10] UNDER VARIOUS EMBEDDING CAPACITIES

Image	PSNR values							
	1,024 bits		4,096 bits		8,192 bits		16,384 bits	
	DiaHyEd	H74CGC	DiaHyEd	H74CGC	DiaHyEd	H74CGC	DiaHyEd	H74CGC
Cameraman	69.9508	77.5360	66.7920	71.5580	63.6830	68.6210	60.8129	65.5320
F16	69.9307	77.5210	66.5968	71.5220	63.5310	68.5940	61.0387	65.5420
Goldhill	69.5378	77.5650	66.5459	71.6090	63.5633	68.5810	60.5774	65.5560
Lena	69.5470	77.6080	66.6624	71.5770	63.5888	68.6140	60.6019	65.5620
Mandrill	69.7060	77.5790	66.6342	71.5580	63.7044	68.5550	60.6101	65.5400
Peppers	69.6211	77.5790	66.6061	71.5080	63.5981	68.5610	60.5426	65.5810

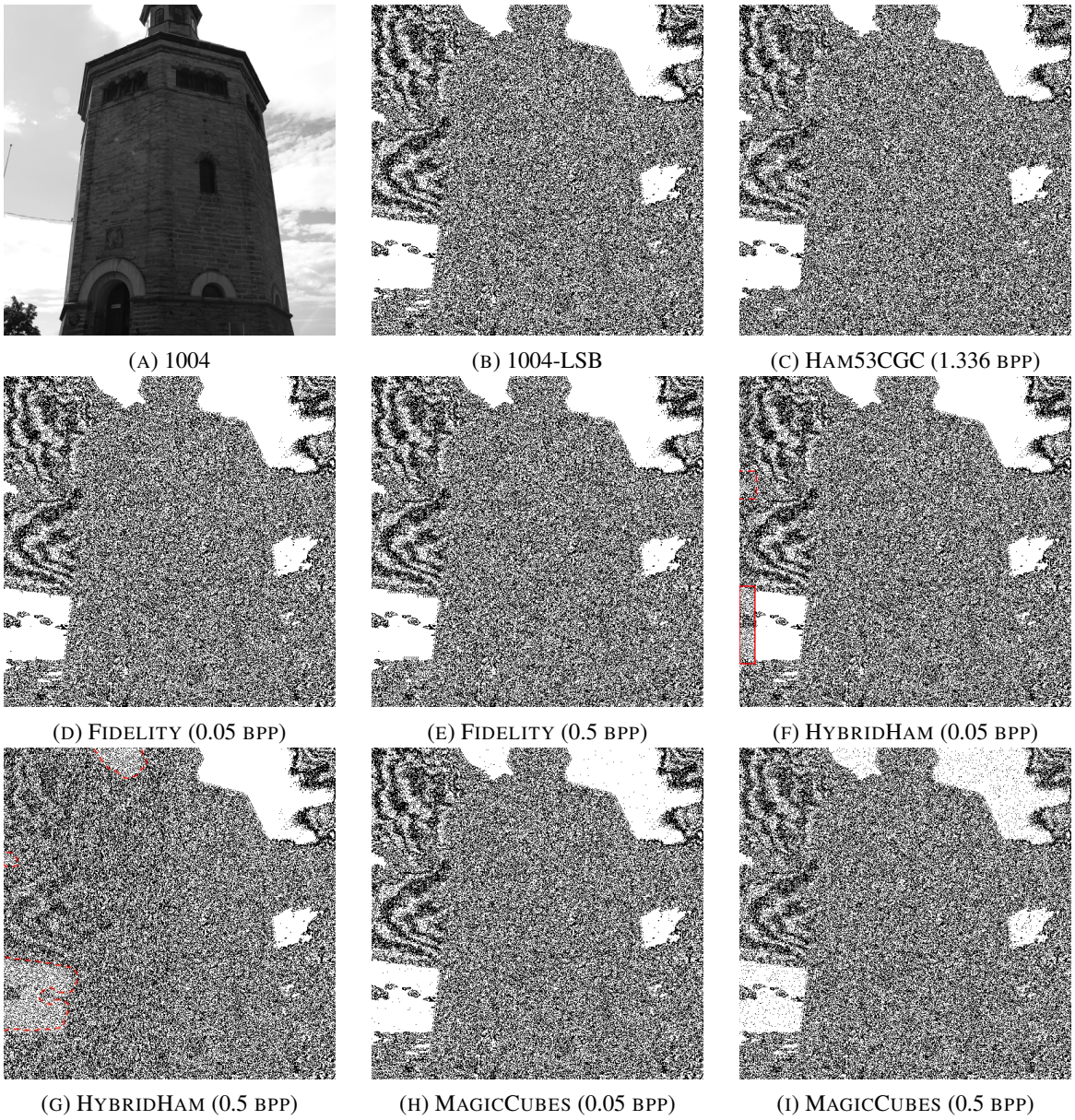
TABLE V. AVERAGE SSIM VALUES OBTAINED FROM 2,000 COVER IMAGES AND THEIR CORRESPONDING STEGO IMAGES GENERATED BY THE PROPOSED SCHEME AND PREVIOUS STEGANOGRAPHY METHODS

ER (bpp)	SSIM values						
	H74PBC	H74CGC	H53PBC	H53CGC	MagicCubes [29]	Fidelity [14]	HybridHam [30]
0.05	0.9999	0.9999	0.9999	0.9999	0.9997	0.9998	0.9998
0.10	0.9998	0.9998	0.9998	0.9998	0.9993	0.9996	0.9995
0.15	0.9997	0.9997	0.9996	0.9996	0.9990	0.9994	0.9992
0.20	0.9995	0.9994	0.9995	0.9994	0.9986	0.9992	0.9990
0.25	0.9993	0.9988	0.9993	0.9990	0.9983	0.9989	0.9987
0.30	0.9989	0.9980	0.9990	0.9985	0.9979	0.9986	0.9984
0.35	0.9983	0.9972	0.9986	0.9977	0.9976	0.9982	0.9982
0.40	0.9974	0.9963	0.9981	0.9969	0.9972	0.9976	0.9979
0.45	0.9966	0.9953	0.9973	0.9960	0.9969	0.9970	0.9976
0.50	0.9957	0.9942	0.9965	0.9951	0.9965	0.9964	0.9974

analysis is used to evaluate the security provided by the proposed scheme. In this steganalysis method, the corresponding pixel is set to 255 if its corresponding LSB is 1; otherwise, its value is kept as the original. As a result of LSB Enhancement, the 1 bits will be highlighted, including the bits that are changed on the LSB layer of the image being carried. Therefore, it is easy to identify the impact of the embedding process with the naked eye. In this experiment, Xstegsecret [31], which is a steganalysis software that detects hidden information from various digital media sources, is implemented to perform LSB Enhancement on the stego images.

Figure 11 illustrates the resulting LSBs of cover image 1004 from the Bows-2 image library obtained using various data hiding techniques. Subfigure (b) presents the LSB bit-plane of the original image, while (c) shows the enhanced LSB of the stego-image generated using Hamming (5, 3) encoding with an embedding payload of 1.336 bpp. Subfigures (d) and (e) depict the enhanced LSBs obtained from the high-fidelity steganography approach [14] at embedding rates of 0.05 bpp and 0.5 bpp, respectively. Similarly, (f) and (g) show the LSBs of the stego-images created by the HybridHam method at the same embedding rates. Finally, (h) and (i) present the LSBs of the stego-images produced by the MagicCubes technique.

In Figure 11, it is evident that the stego image generated by the proposed scheme is resilient against the LSB Enhancement attack. This robustness is achieved by refraining from utilizing smooth image regions, such as the sky, or highly uniform areas, like house walls, even



(A) 1004 (B) 1004-LSB (C) HAM53CGC (1.336 BPP)

(D) FIDELITY (0.05 BPP) (E) FIDELITY (0.5 BPP) (F) HYBRIDHAM (0.05 BPP)

(G) HYBRIDHAM (0.5 BPP) (H) MAGICCUBES (0.05 BPP) (I) MAGICCUBES (0.5 BPP)

Figure 11. Cover image 1004 from Bows-2 (A), LSB bit-plane (B), stego images generated by various methods at different embedding rates (C-I)

when the embedding rate is 1.336 bpp. In contrast, the resulting LSB (Figure 11F), produced by the LSB Enhancement method, of the stego image yielded by the HybridHam consists of additional patterns in the image areas that were originally flat regions in the cover image (marked by red rectangles), even when the embedding capacity is only 0.05 bpp. Similarly, in the resulting LSBs of the stego image (Figure 11H and Figure 11I) produced by MagicCubes method, there is a noise like pepper noise added to the smooth image regions, including the sky and the flat regions.

Furthermore, to clarify how the complexity threshold affects the selection of edge pixels for information concealment, a bit modification map of pixel values between the original image and the stego-image is generated. This map highlights the modification points after message embedding and is shown in Figure 12. As illustrated, the modification map of the stego-image produced by the proposed scheme more accurately reflects image complexity compared to previous methods. The locations of the embedded pixels are concentrated in complex regions of the image, such as the wall of the tower, tree branches, and the non-uniform cloud areas. Similarly, the Fidelity method exploits the pixels that lie on high complex image regions for hiding the message bits. Therefore, the modification map of

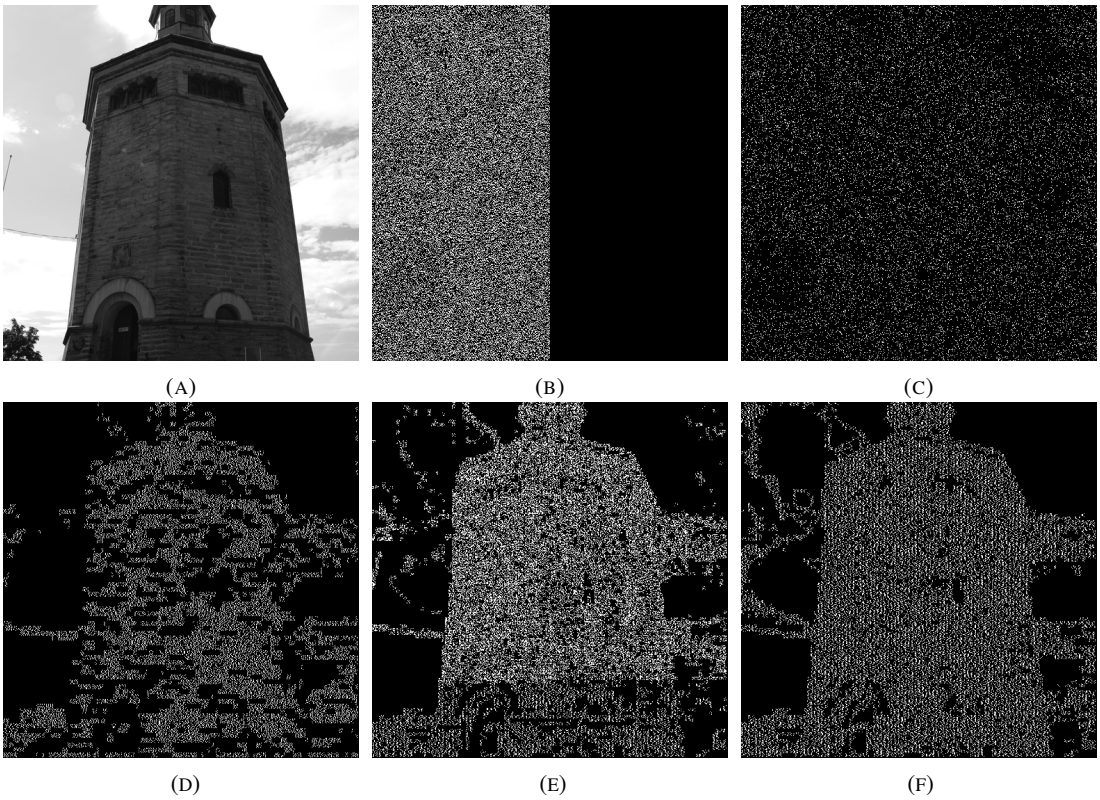


Figure 12. The LSB bit-plane of cover image 1004 (A) and the bit modification map generated by hybridham (B), magiccubes (C), fidelity (D), H53CGC (E), and H53PBC (F)

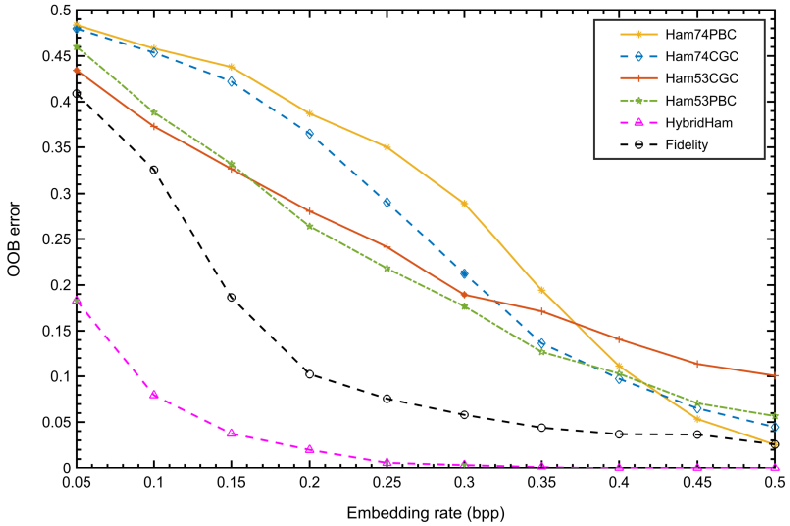


Figure 13. Out-of-bag (OOB) error distribution for 2,000 stego images from the proposed hamming code-based methods and existing approaches

it expresses that the stego-pixels belong to complex image areas. In contrast, the HybridHamming and MagicCubes methods do not consider block complexity in data hiding. Consequently, their bit modification maps reveal modification patterns spread across most image regions, thereby reducing resistance to detection attacks such as LSB enhancement.

2. *Security Against Ensemble Classifier-Based Steganalysis*

To evaluate the security of the proposed scheme for different image texture characteristics, an ensemble classifier is utilized instead of the more commonly used support vector machine (SVM) [32]. This method works by combining the predictions of multiple base learners, which

improves the overall accuracy of the classifier by reducing the impact of any individual base learner's errors [33]. During the training process, some observations are left out or "out-of-bag" (OOB) for each tree. If the OOB error value is close to 0.5, the ensemble-based steganalysis is failed to detect the existence of the hidden message in the stego images.

In this experiment, the second-order Subtractive Pixel Adjacency Matrix (SPAM) features extractor [34] is utilized to extract feature characteristic from the cover images and their corresponding stego images. Subsequently, the feature sets extracted from the stego images, generated by the variants of the proposed scheme, including Ham53CGC, Ham53PBC, Ham74CGC, Ham74PBC, as well as the two previous approaches, Hybrid Hamming and Fidelity, are used as stego feature sets. For each set of extracted stego SPAM features, the evaluation process is performed 5 times and the result is taken as the average of the 5 executions. As the obtained OOB values illustrated in the Fig. 13, it is evident that the proposed scheme is superior to the previous approach against ensemble classifier steganalysis even at the high embedding rates.

Since the embedding modifications are concentrated in complex regions, the introduced changes are masked by the inherent noise and texture patterns. This reduces the statistical detectability of embedding traces in the feature space, leading to weaker discrimination capability of the base learners. As a result, the ensemble classifier exhibits higher OOB error. Similar to the proposed scheme, Fidelity method [14] considers the texture characteristics by employing Canny edge detection algorithm to identify the sharpness degree of a pixel block to hide message bits. To reduce the degradation of visual quality, Hamming codes-based data hiding method is employed. Nevertheless, the OOB error of it is still higher than that of the proposed scheme for different embedding rates. The primary reason for the high OOB error is that Fidelity method does not consider the textured complexity of pixel blocks for each used bit-plane as performed in the proposed scheme.

On the contrary to the proposed scheme, HybridHam [30] method employs Hamming code to hide message bits and optimal pixel adjustment process for minimizing the distortion. However, this method does not consider the characteristic of image regions, yielding strong statistical artifacts in the residual features, making cover and stego images more distinguishable. Consequently, the ensemble classifier achieves the lowest OOB error in comparison with the proposed scheme and Fidelity.

3. Security Against Convolutional Neural Network-Based Steganalysis (XuNet)

To further examine security against detection attacks, a steganalysis method based on convolutional neural networks, namely XuNet [35], is employed. In this experiment, 10,000 images from the Bows-2 database are sequentially embedded with data at 0.1 bpp, 0.5 bpp, and 1.0 bpp. Subsequently, 5,000 pairs of cover and stego images are randomly selected to form the testing set. The remaining 5,000 pairs, consisting of cover images and their corresponding stego images, are used for training. Within this set, 4,000 pairs are employed for training and 1,000 pairs for validation.

TABLE VI. ACCURACY (%) OF XU-NET AGAINST THE PROPOSED SCHEME

	0.1 bpp	0.5 bpp	1.0 bpp
Proposed	49.97	53.90	62.83

The experiment follows the hyperparameter settings of the XuNet project [36], with a learning rate of 0.001 decreased by 10% every 5,000 iterations and a mini-batch size of 64.

From Table VI, the accuracy of the CNN-based steganalysis method XuNet increases from 53.90% to 62.83% at an embedding rate (ER) of 1.0 bpp, indicating that it remains ineffective in distinguishing cover images from stego images generated by the proposed scheme. In particular, at an embedding rate of 0.1 bpp, XuNet achieves an accuracy of 49.97%, which is nearly equivalent to random guessing between cover and stego images.

The primary reason for this is that the adaptive embedding strategy effectively utilizes high-texture image regions to conceal the message bits, resulting in the minimization of visual distortion in stego-images. Furthermore, the number of bit flips required is further reduced by employing Hamming codes-based data hiding methods. Consequently, the

degradation caused by data hiding is significantly less than the amount that XuNet steganalysis is capable of detecting the presence of the hidden information within stego-images.

D. Analysis of Computational Complexity

In this section, the computational complexity of the proposed scheme is briefly analyzed. The approach consists of two major processes: complexity threshold estimation and data hiding. These two processes are performed sequentially; therefore, the overall computational complexity of the proposed scheme is determined by the process that requires more computing resources.

The primary operation in complexity threshold estimation is counting the number of pixel blocks whose complexity is sufficient to completely embed the given data in the cover image. The complexity of a pixel block is defined as the sum of the absolute differences between each pixel and its 8 neighboring pixels. As a result, the computational cost of estimating the complexity of each pixel block is $\mathcal{O}(1)$.

Given a cover image of size $WS \times HS$, divided into $nBlbk$ blocks of size $mB \times nB$, where WS and HS denote the image width and height, and mB and nB represent the number of columns and rows of each block, the overall complexity of threshold estimation is $\mathcal{O}(nBlbk)$.

The second primary process of the proposed scheme is data hiding. Its elementary operation involves determining the embedding solutions from the input pixel blocks. In other words, this process identifies the pixel positions whose LSBs need to be flipped to hide the given message bits. The data-bit embedding operation is implemented through matrix multiplication of two arrays: the message bits and the cover bits (selected from a specific bit-plane of the cover image). Since the sizes of these arrays are relatively small, the computational complexity of the operation is $\mathcal{O}(1)$. In certain cases, the primary operations of data hiding are performed kL times to embed the entire set of secret bits. Therefore, the running time of the embedding procedure is $\mathcal{O}(kL \times nBlbk)$. Consequently, the proposed scheme is suitable for real-time applications.

V. Conclusion

This paper proposes a novel secure image steganography scheme that employs a Hamming code-based data hiding approach. In this method, block complexity estimation based on the zig-zag order scanning technique is developed to select textured image regions for hiding message bits. This technique helps enhance the security of the hidden message against both statistical steganalysis and LSB enhancement analysis approaches. Experimental results have substantiated that the proposed scheme can effectively embedding varying size payload with a significantly higher embedding rate and better image quality compared to the relevant approaches. However, there are several image blocks with complexity levels below the threshold after the message has been embedded. Although the number of such pixel blocks is negligible and does not affect the perceptual quality of the carrier image or the security of the hidden confidential data, our future work is focused on finding a better solution to address this disadvantage.

Acknownledgment

We appreciate the assistance of our colleagues and the financial support provided by School Of Interdisciplinary Sciences and Arts, Vietnam National University, Hanoi.

REFERENCES

[1] V. T. Son and T. M. Duc, "A robust watermarking method based on rdwt-dct-svd in the ycbcr color space," *Journal of Science and Technology on Information security*, pp. 5–14, Dec. 2024. DOI: http://dx.doi.org/10.54654/isj.v3i23.1054

[2] D. Grībermans, A. Jeršovs, and P. Rusakovs, "Development of requirements specification for steganographic systems," *Applied Computer Systems*, vol. 20, no. 1, pp. 40–48, Dec. 2016. DOI: http://dx.doi.org/10.1515/acss-2016-0014

[3] S. Kumar, A. Singh, and M. Kumar, "Information hiding with adaptive steganography based on novel fuzzy edge identification," *Defence Technology*, vol. 15, no. 2, pp. 162–169, Apr. 2019. DOI: http://dx.doi.org/10.1016/j.dt.2018.08.003

[4] B. Ray, S. Mukhopadhyay, S. Hossain, S. K. Ghosal, and R. Sarkar, "Image steganography using deep learning based edge detection," *Multimedia Tools and Applications*, vol. 80, no. 24, pp. 33 475–33 503, Aug. 2021. DOI: http://dx.doi.org/10.1007/s11042-021-11177-4

[5] K. Gaurav and U. Ghanekar, "Image steganography based on canny edge detection, dilation operator and hybrid coding," *Journal of Information Security and Applications*, vol. 41, pp. 41–51, Aug. 2018. DOI: http://dx.doi.org/10.1016/j.jisa.2018.05.001

[6] S.-Y. Shen and L.-H. Huang, "A data hiding scheme using pixel value differencing and improving exploiting modification directions," *Computers & Security*, vol. 48, pp. 131–141, Feb. 2015. DOI: http://dx.doi.org/10.1016/j.cose.2014.07.008

[7] H. Al-Dmour and A. Al-Ani, "A steganography embedding method based on edge identification and xor coding," *Expert Systems with Applications*, vol. 46, pp. 293–306, Mar. 2016. DOI: http://dx.doi.org/10.1016/j.eswa.2015.10.024

[8] W.-J. Chen, C.-C. Chang, and T. H. N. Le, "High payload steganography mechanism using hybrid edge detector," *Expert Systems with Applications*, vol. 37, no. 4, pp. 3292–3301, Apr. 2010. DOI: http://dx.doi.org/10.1016/j.eswa.2009.09.050

[9] S. Islam, M. R. Modi, and P. Gupta, "Edge-based image steganography," *EURASIP Journal on Information Security*, vol. 2014, no. 1, Apr. 2014. DOI: http://dx.doi.org/10.1186/1687-417x-2014-8

[10] D. R. I. M. Setiadi, "Improved payload capacity in lsb image steganography uses dilated hybrid edge detection," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 2, pp. 104–114, Feb. 2022. DOI: http://dx.doi.org/10.1016/j.jksuci.2019.12.007

[11] C. Kim and C.-N. Yang, "Data hiding based on overlapped pixels using hamming code," *Multimedia Tools and Applications*, vol. 75, no. 23, pp. 15 651–15 663, Nov. 2014. DOI: http://dx.doi.org/10.1007/s11042-014-2355-x

[12] B. Jana, D. Giri, and S. Kumar Mondal, "Dual image based reversible data hiding scheme using (7,4) hamming code," *Multimedia Tools and Applications*, vol. 77, no. 1, pp. 763–785, Jan. 2017. DOI: http://dx.doi.org/10.1007/s11042-016-4230-4

[13] C. Kim, D. Shin, C.-N. Yang, and Y.-S. Chou, "Improving capacity of Hamming (n,k) + 1 stego-code by using optimized Hamming + k," *Digital Signal Processing*, vol. 78, pp. 284–293, Jul. 2018. DOI: http://dx.doi.org/10.1016/j.dsp.2018.03.016

[14] C.-F. Lee, C.-C. Chang, X. Xie, K. Mao, and R.-H. Shi, "An adaptive high-fidelity steganographic scheme using edge detection and hybrid hamming codes," *Displays*, vol. 53, pp. 30–39, Jul. 2018. DOI: http://dx.doi.org/10.1016/j.displa.2018.06.001

[15] W. Luo, F. Huang, and J. Huang, "Edge adaptive image steganography based on lsb matching revisited," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 201–214, Jun. 2010. DOI: http://dx.doi.org/10.1109/tifs.2010.2041812

[16] E. Kawaguchi and R. O. Eason, "Principles and applications of bpcs steganography," in *Multimedia systems and applications*, vol. 3528. SPIE, 1999, pp. 464–473.

[17] S. Sun, "A new information hiding method based on improved bpcs steganography," *Advances in Multimedia*, vol. 2015, pp. 1–7, 2015. DOI: http://dx.doi.org/10.1155/2015/698492

[18] F. Andrieux and C. Deltel, "Jpeg steganography using bpcs," in *Proceedings of the 20th Annual SIG Conference on Information Technology Education*, ser. SIGITE '19. ACM, Sep. 2019, pp. 163–163. DOI: http://dx.doi.org/10.1145/3349266.3351383

[19] V. Sabeti, S. Samavi, and S. Shirani, "An adaptive lsb matching steganography based on octonary complexity measure," *Multimedia Tools and Applications*, vol. 64, no. 3, pp. 777–793, Jan. 2012. DOI: http://dx.doi.org/10.1007/s11042-011-0975-y

[20] C. Munuera, "Hamming codes for wet paper steganography," *Designs, Codes and Cryptography*, vol. 76, no. 1, pp. 101–111, 2014. DOI: http://dx.doi.org/10.1007/s10623-014-9998-5

[21] Y. Kim, Z. Duric, and D. Richards, *Modified Matrix Encoding Technique for Minimal Distortion Steganography*. Springer Berlin Heidelberg, 2007, pp. 314–327. DOI: http://dx.doi.org/10.1007/978-3-540-74124-4_21

[22] F. Huang, J. Huang, and Y.-Q. Shi, "New channel selection rule for jpeg steganography," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1181–1191, Aug. 2012. DOI: http://dx.doi.org/10.1109/tifs.2012.2198213

[23] Y. Zhong, F. Huang, and D. Zhang, *New Channel Selection Criterion for Spatial Domain Steganography*. Springer Berlin Heidelberg, 2013, pp. 1–7. DOI: http://dx.doi.org/10.1007/978-3-642-40099-5_1

[24] S. Mozaffari, "Parallel image encryption with bitplane decomposition and genetic algorithm," *Multimedia Tools and Applications*, vol. 77, no. 19, pp. 25 799–25 819, Apr. 2018. DOI: http://dx.doi.org/10.1007/s11042-018-5817-8

[25] T. D. Nguyen and H. D. Le, "A reversible data hiding scheme based on (5, 3) hamming code using extra information on overlapped pixel blocks of grayscale images," *Multimedia Tools and Applications*, vol. 80, no. 9, pp. 13 099–13 120, Jan. 2021. DOI: http://dx.doi.org/10.1007/s11042-020-10347-0

[26] A. Saeed, Fawad, M. J. Khan, H. Shahid, S. I. Naqvi, M. A. Riaz, M. S. Khan, and Y. Amin, "An accurate texture complexity estimation for quality-enhanced and secure image steganography," *IEEE Access*, vol. 8, pp. 21 613–21 630, 2020. DOI: http://dx.doi.org/10.1109/access.2020.2968217

[27] D. Laishram and T. Tuithung, "A novel minimal distortion-based edge adaptive image steganography scheme using local complexity: (beass)," *Multimedia Tools and Applications*, vol. 80, no. 1, pp. 831–854, Sep. 2020. DOI: http://dx.doi.org/10.1007/s11042-020-09519-9

[28] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004. DOI: http://dx.doi.org/10.1109/tip.2003.819861

[29] J. J. Ranjani and F. Zaid, "Pseudo magic cubes: A multidimensional data hiding scheme exploiting modification directions for large payloads," *Computers & Electrical Engineering*, vol. 89, p. 106928, Jan. 2021. DOI: http://dx.doi.org/10.1016/j.compeleceng.2020.106928

[30] C. Kim, D. Shin, B.-G. Kim, and C.-N. Yang, "Secure medical images based on data hiding using a hybrid scheme with the hamming code, lsb, and opap," *Journal of Real-Time Image Processing*, vol. 14, no. 1, pp. 115–126, Feb. 2017. DOI: http://dx.doi.org/10.1007/s11554-017-0674-7

[31] A. Muñoz, "Stegsecret. a simple steganalysis tool." , (2007), accessed: 22/9/2025, *http://stegsecret.sourceforge.net/*.

[32] J. Kodovsky, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, Apr. 2012. DOI: http://dx.doi.org/10.1109/tifs.2011.2175919

[33] Q. Li, G. Feng, H. Wu, and X. Zhang, *Ensemble Steganalysis Based on Deep Residual Network*. Springer International Publishing, 2020, pp. 84–95. DOI: http://dx.doi.org/10.1007/978-3-030-43575-2_7

[34] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, Jun. 2010. DOI: http://dx.doi.org/10.1109/tifs.2010.2045842

[35] G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, May 2016. DOI: http://dx.doi.org/10.1109/LSP.2016.2548421

[36] S. Brijesh, "XuNet: Structural design of convolutional neural networks for steganalysis," (2016), Accessed: 2025-08-26, *https://github.com/brijeshiitg/XuNet-Structural-Design-of-Convolutional-Neural-Networksfor-Steganalysis/*.

About the authors

Nguyen Duc Tuan
Workplace: Vietnam National University, Hanoi, School of Interdisciplinary Sciences and Arts
Email: tuannguyenduc@vnu.edu.vn
Education: received the PhD degree in Computer Science from Khon Kaen University (KKU), Khon Kaen, Thailand, in 2016. He received the M.SC degree from Le Qui Don Technical Univeristy, Vietnam in 2008.
Recent research interests: Cryptography, Steganography, Reversible Data Hiding, Blockchain, Data Science, IoT

Tên tác giả: **Nguyễn Đức Tuấn**
Cơ quan công tác: Trường Khoa học Liên ngành và Nghệ thuật, Đại học Quốc Gia Hà Nội
Email: tuannguyenduc@vnu.edu.vn
Quá trình đào tạo: Nhận bằng Tiến sĩ Khoa học máy tính tại Đại học Khon Kaen, Khon Kaen, Thái Lan năm 2016; Nhận bằng Thạc sĩ Khoa học máy tính tại Trường Đại học Kỹ thuật Lê Quý Đôn năm 2008

Hướng nghiên cứu hiện nay: Mật mã, viết phủ, giấu tin có thể hồi phục, công nghệ chuỗi khối, khoa học dữ liệu, Internet vạn vật.