http://infor.seaninstitute.org/index.php/infokum/index
JURNAL INFOKUM, Volume 9, No.2, Juni 2021 ISSN : 2302-9706

Capacity Optimization On RGB Overlapping Block-Based Pixel Value Differencing Image Steganography With Adaptive Threshold

Rosyidah Siregar [1], Tommy [2], Andi Marwan Elhanafi [3], Nenna Irsa Syahputri [4], Manovri Yeni [5]
[1,2,3,4] Universitas Harapan Medan, [5] Universitas Muhammadiyah Aceh
rosyidah_siregar.unhar@harapan.ac.id , tomshirakawa@gmail.com , andimarwanelhanafi@gmail.com , nenna_irsa.unhar@harapan.ac.id , manovri.yeni@unmuha.ac.id

Abstract

Article Info
Received 01 June 2021
Revised 20 June 2021
Accepted 30 June 2021

Pixel value differencing steganography is an image steganography that utilizes the difference of the image pixel value to embed the secret message bits. RGB overlapped block-based PVD was introduced by Prasad and Pal which uses the difference value in the pair of RGB color components of a pixel compared to using the difference value of two consecutive pixels. This approach has good performance at increasing capacity especially in images with low pixel variance values. The RGB overlapped block-based PVD algorithm uses a threshold that limits the amount of difference in the color component pairs that are allowed to embed the secret message bits. The use of a global threshold will reduce the potential for optimal capacity utilization of the container image. This study implements an adaptive threshold that uses two different types of thresholds that use the embedding bit limit and the RMSE difference of the pixels before and after the embedding process to the next pixel. This optimization is able to provide a better capacity increase with PSNR degradation from the previous algorithm which is quite low.

Keywords: steganography, PVD, RGB, overlapping, threshold

1. Introduction

Steganography can be defined as a field of science that involves the communication of secret messages using multimedia media such as images, video and audio [1]. The use of multimedia media in the field of steganography has become a common thing considering that a small change that occurs in the media will not have a significant impact on the representation of the media. Digital image is one of the media that is very often used in the implementation of steganography. The high level of tolerance of human vision to slight changes that occur in digital images is one of the reasons for the many research and development of steganographic methods using digital images. In addition, the high pixel redundancy in the digital image makes it a better option for hiding confidential information [2].

Steganography algorithms in digital images are generally divided into two types, namely spatial-domain and transform-domain. The Spatial-domain algorithm works by hiding confidential information by making changes to the brightness or chrominance values in the container image. Meanwhile, the transform-domain algorithm works by hiding secret information in the transformed signal from a digital image [3]. Spatial-domain steganography is easier to implement than transform-domain steganography, this is because the operation of hiding secret information can be carried out directly on the value of the pixels contained in the container image compared to the transform-domain which requires the transformation process to the

pixel values of the image container. Some simple examples of spatial-domain implementations such as least significant bit substitution (LSB) are widely used in the implementation and research of digital image steganography [4][5]. The development of LSB in digital image steganography is also continuously carried out such as increasing capacity using Multiple Least Significant Bit [6], optimizing image quality of LSB implementation [7], inverted LSB [8] and modified LSB [9].

Pixel Value Differencing Algorithm is a digital image steganography algorithm that uses the difference value in pixels in the process of hiding secret information. PVD was first proposed by Wu and Tsai [10] who found that the edge area of an object in an image can be used to accommodate more secret data because there is a large difference in pixel values in that area. The difference in value can be used as a substitution value with the bit of secret information. Over time, many developments have been made to the PVD algorithm. There are several motivations in developing PVD methods such as capacity building [11], adaptive block PVD [12] [13] and pixel overlapping PVD [14].

Prasad and Pal [15] proposed a PVD steganography model that uses the difference in the color component values in RGB as a substitution reference or can be called RGB overlapping block-based. Unlike the conventional PVD algorithm which uses the difference values at two or more pixels, the algorithm proposed by Prasad and Pal embed the secret bits on each pixel independently without being influenced by other pixels. The basic concept of this method is to form pairs consisting of color components, namely (R,G) and (G,B). The difference value needed in the process of hiding information in PVD steganography is obtained from the color component pairs. This method provides a better capacity increase compared to conventional PVD, especially in images with low pixel variance values. However, by not involving the value of the surrounding pixels, there is a large change due to the substitution process can cause considerable distortion in the resulting image. To minimize this distortion, the proposed method uses a threshold that limits the data hiding capacity to maintain image quality. The use of a threshold is able to limit the distortion that occurs in the resulting image, but also eliminates the potential for a larger capacity that may exist in the container image. This study will modify the block-based RGB overlapping method using adaptive thresholds that consist of capacity threshold and quality threshold obtained by comparing the root mean square error (RMSE) of the pixel being processed with the consecutive pixel before and after the data hiding process. By using this approach, the limitation of data hiding capacity is not carried out globally, but depends on the distortion that occurs in pixels after the data hiding process so that the potential of the capacity of the container image can be optimized.

2. Method

2.1 Pixel Value Differencing

Pixel Value Differencing as proposed by Wu and Tsai [10] uses the grayvalue of two adjacent pixels. A pixel will be used in the data hiding process if the difference between that pixel and the next pixel meets the specified range. pixels with a small difference value or called the smooth region will accommodate fewer secret message bits compared to pixels with a large difference to minimize distortion of the resulting image. Pixels in the image are divided into groups of two consecutive (1 x 2) pixels where each group cannot use pixels from other groups (non-overlapping). Two pixels in the i-th group are denoted as P_i dan P_{i+1} and the distance between the two pixels is denoted by d_i which is obtained from the absolute difference of the gray values P_i and P_{i+1}.

$$d_i = |P_i - P_{i+1}| \quad (1)$$

The number of secret message bits that will be embed in the pixel group depends on the difference or distance value of the pixel pair. Conventional PVD uses a quantization table obtained from the grayvalues (0 - 255) into several divisions of range (R) as shown in table 1. Each range (R_i) has a lower limit (l_i) and an upper limit (u_i). The greater the value of the range, the greater the bits of the secret message that can be embedded. The number of bits (t) that can be inserted can be calculated using equation 2.

$$t = \lfloor log_2(u_i - l_i + 1) \rfloor \quad \text{.....(2)}$$

Table 1. PVD Quantization Range

Range	R1	R2	R3	R4	R5	R6
Lower	0	8	16	32	64	128
Upper	7	15	31	63	127	255
Width	8	8	16	32	64	128
Bit Capacity	3	3	4	5	6	7

The process of embedding the secret message bits begins by calculating the distance d_i from the pixel pairs obtained from reading the pixel image of the container. Using the obtained distance, the lower limit of u_i and the upper limit l_i of the range R_i are define by finding the corresponding R_i where the value of d_i is between its upper and lower limit. For example, if the value of d is 12, then the corresponding range is R_2because 12 is between the lower limit (8) and the upper limit (15) of R_2, so that the lower limit of 8 and the limit of 15 are obtained. The next process is to calculate the number of bits that can be inserted in the pixel pair (t). After the number of bits to be embedded is calculated, the new distance d_i' can be calculated using equation 3 which is then continued by calculating the value of the difference (m) between the old distances and new distances using equation 4.

$$d_i' = l_i + Secret\ Message\ Bits\ (Decimal) \quad \text{.....(3)}$$

$$m = |d_i' - d_i| \quad \text{.....(4)}$$

Using the value of m, update is made from the pixel values P_i and P_{i+1} to P_i' and P_{i+1}' as the result of the embedding process using the following conditions:

$$(P_i', P_{i+1}') = \begin{cases} \left(P_i + \left\lceil \frac{m}{2} \right\rceil, P_{i+1} - \left\lfloor \frac{m}{2} \right\rfloor\right), if\ P_i \geq P_{i+1}\ and\ d_i' > d_i \\ \left(P_i - \left\lfloor \frac{m}{2} \right\rfloor, P_{i+1} + \left\lceil \frac{m}{2} \right\rceil\right), if\ P_i < P_{i+1}\ and\ d_i' > d_i \\ \left(P_i - \left\lfloor \frac{m}{2} \right\rfloor, P_{i+1} + \left\lceil \frac{m}{2} \right\rceil\right), if\ P_i \geq P_{i+1}\ and\ d_i' \leq d_i \\ \left(P_i + \left\lceil \frac{m}{2} \right\rceil, P_{i+1} - \left\lfloor \frac{m}{2} \right\rfloor\right), if\ P_i < P_{i+1}\ and\ d_i' \leq d_i \end{cases} \quad \text{.....(5)}$$

P_i' dan P_{i+1}'' represent the pixel value resulting from the embedding process which can then be used in the extraction process where the decimal value of the embedded secret message bit can be obtained by

calculating the value of the pixel difference that has been inserted and then subtracting the lower limit of the range in which the difference lies.

$$Secret\ Message\ Bits\ (Decimal) = l_i - |P_i' - P_{i+1}'| \quad \text{.....(6)}$$

2.2 RGB Overlapping Block Based PVD

RGB overlapping block based uses the difference of the color component pairs (R,G) and (G,B) where the G component acts as an overlapping block [15]. The PVD process is applied to each pair (R, G) and (G, B) following the conventional PVD process, only the value that is operated is the value of the pixel color components. After the PVD process is applied to (R,G) and (G,B) pairs, new values will be obtained for each color component, namely (R_1, G_1) and (B_1, G_2). The results of this stage will produce two different G values, namely G_1 and G_2 as a result of different PVD operations performed on R and B components. The final G value G_s is obtained by calculating the average of G_1 and G_2. Once the final G value is obtained (G_s) then the R_1 and B_1 values will be adjusted to produce the final R and B values (R_s, B_s).

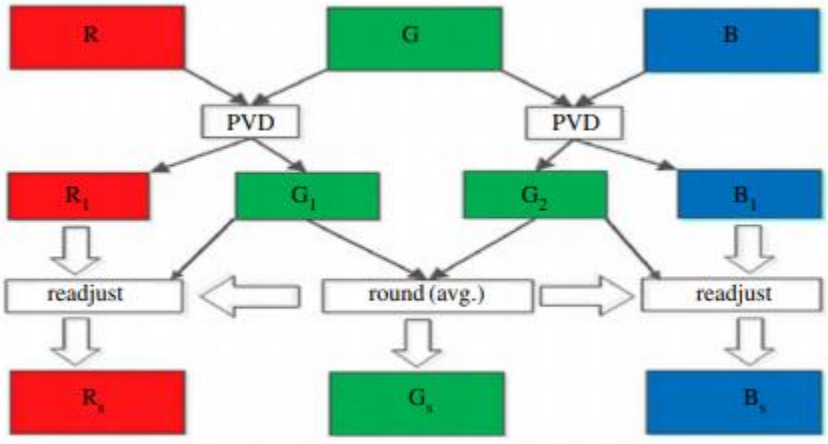


Figure 1. Schematic diagram of embedding procedure [15]

The insertion procedure of RGB overlapped block-based pvd can be described as follows:

1. Read a RGB pixel from color cover image and decompose it into R, G and B respectively.
2. Form two pairs like (R, G) and (G, B).
3. Compute t1 = |R − G| and t2 = |G − B|
4. If (t1 + t2) < Threshold Execute Step 5 to 7
5. Apply pixel value differencing (PVD) in both (R,G) and (G,B) pairs to embed the secret message bits.
6. Get intermediate stego color components
   a. R_1 and G_1 from (R, G) pair
   b. G_2 and B_1 from (G, B) pair
7. Perform readjustment process to form red, green and blue color stego components based on the following sub steps:
   a. Compute $G_{average} = Round\left(\frac{G_1+G_2}{2}\right)$ .....(7)
   b. Modify R_1 as final stego red color component

$$R_S = R_1 - (G_1 - G_{average}) \quad \text{.....(8)}$$

c. Compute final stego green color component

$$G_S = G_{average} \quad \text{.....(9)}$$

d. Modify B_1 as final stego green color component

$$B_S = B_1 - (G_2 - G_{average}) \quad \text{.....(10)}$$

8. Process rest of the color pixels using step 1 to 7.

2.3 Proposed Optimization

RGB overlapping block based using the threshold value obtained from the accumulated distance between pairs (R,G) and (G,B). Using a small threshold value will reduce the capacity of the pixels to be inserted. While using a large threshold value will increase the capacity of the pixels to be inserted but will also cause considerable distortion in the resulting image. A sufficiently large threshold value will also cause overflow/underflow problems to occur in the value of the inserted color component and additional complex adjustments are needed considering that adjustments to one color component will have an impact on the other two color components or pixels that experience overflow/underflow are ignored which results in a decrease usable capacity of the container image.

Choosing the right threshold value is a very difficult thing to do considering the color characteristics of the image are different from one to another. This study modifies the processing stages of the RGB overlapping block based by adding an adaptive threshold to optimize the capacity of the container image pixels and maintain the quality of the resulting image.

The adaptive thresholds used in this study is the use of two different thresholds depending on the final value of the color component obtained. If the previous RGB overlapping block based used a threshold value in the form of the maximum allowed distance from the accumulated distance pairs (R, G) and (G, B), then in this study a threshold consisting of two thresholds, namely the maximum number of bits that can be inserted in each – each pair (R,G) and (G,B) and the RMSE threshold between the original and embedded pixel to the next pixel. If the bit capacity of the secret message that can be embedded is greater than the specified threshold, the amount of distortion will be calculated by comparing the root mean square error (RMSE) between the initial pixel and the final pixel to the next pixel. If the RMSE difference obtained is below the specified RMSE threshold, then the embedding process will be carried out as in the original RGB overlapping block-based process.

The insertion procedure of the proposed optimization is described as follows:

1. Define the max bit threshold (T_{bits}) and max RMSE threshold (T_{RMSE}).
2. Read a RGB pixel from color cover image and decompose it into R, G and B respectively.
3. Form two pairs like (R, G) and (G, B).
4. Apply pixel value differencing (PVD) in both (R,G) and (G,B) pairs to calculate the capacity of both pairs and to embed the secret message bits.
5. Get intermediate stego color components
   a. R_1 and G_1 from (R, G) pair
   b. G_2 and B_1 from (G, B) pair

6. Perform readjustment process to form red, green and blue color stego components based on the following sub steps:
   a. Compute $G_{average} = Round\left(\frac{G_1+G_2}{2}\right)$ .....(11)
   b. Modify R_1 as the candidate of final stego red color component

      $R_S = R_1 - (G_1 - G_{average})$ .....(12)

   c. Compute candidate of final stego green color component

      $G_S = G_{average}$ .....(13)

   d. Modify B_1 as the candidate of final stego green color component

      $B_S = B_1 - (G_2 - G_{average})$ .....(14)

7. If any candidate of final color component values have overflow/underflow values then ignore the pixels and skip to step 11.
8. If the capacity obtained from (R,G) and (R,B) ≤ max bit threshold (T_{bits}) then continue to step 10.
9. If the capacity obtained from (R,G) and (R,B) > max bit threshold (T_{bits}) then perform the following sub steps :
   a. Calculate RMSE value of the original pixel and consecutive pixel.

$$RMSE_{(P_i,P_{i+1})} = \sqrt{\frac{(P_i[R]-P_{i+1}[R])^2+(P_i[G]-P_{i+1}[G])^2+(P_i[B]-P_{i+1}[B])^2}{2}} \quad .....(15)$$

   b. Calculte RMSE value of embedded pixel and consecutive pixel.

$$RMSE_{(P_i',P_{i+1})} = \sqrt{\frac{(P_i'[R]-P_{i+1}[R])^2+(P_i'[G]-P_{i+1}[G])^2+(P_i'[B]-P_{i+1}[B])^2}{2}} \quad .....(16)$$

   c. Calculate the RMSE difference.

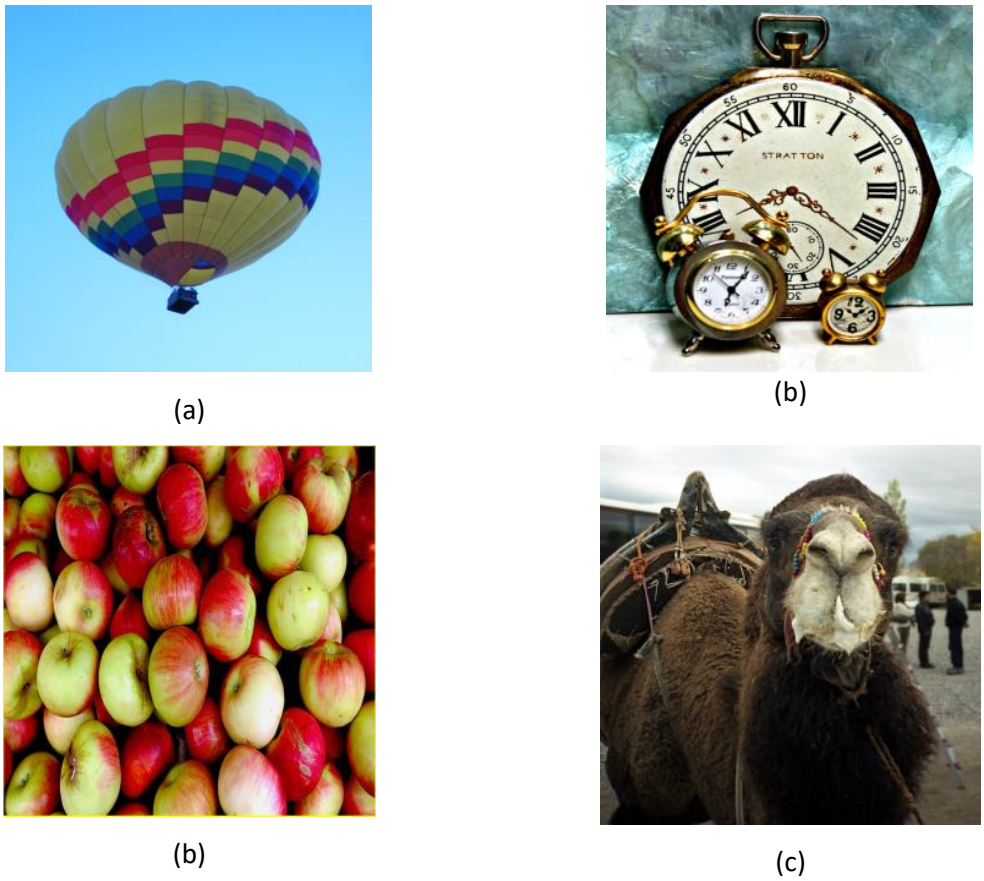
$$RMSE_{diff} = \left|RMSE_{(P_i,P_{i+1})} - RMSE_{(P_i',P_{i+1})}\right| \quad .....(17)$$

   d. If $RMSE_{diff} \leq T_{RMSE}$ then continue to step 10.
   e. If $RMSE_{diff} > T_{RMSE}$ then ignore the pixel and skip to step 11.
10. Apply the candidate final color components (R_S, G_S, B_S) as the new color components of embedded pixel.
11. Process rest of the color pixels using step 1 to 10.

3. Results and Discussion

Tests in this research were carried out using several different color images as container images. The message that is used as a secret message is obtained randomly with the bit length of the message

proportional to the number of pixels from the container image. The tests carried out will analyze the comparison of the quality and capacity between the overlapped block-based RGB images and the optimization one.

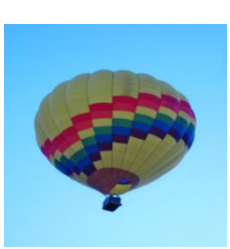


(a) (b)

(b) (c)

Figure 2. Test images

The test images used in this study as shown in Figure 7 was selected randomly which has significant color features differences between images. Each test image will be processed separately using the original RGB overlapped block-based algorithm and the optimization using the adaptive threshold which was developed in this research.



Original Image

Original RGB Overlapped Block-Based
PSNR = 44.5305
Capacity = 41,264

With Adaptive Thresholding
PSNR = 44.2726
Total Kapasitas = 44,635

Figure 3. Ballon image test result

Test on the balloon image as shown in Figure 8 using $T_{bits} = 4$ and $T_{RMSE} = 0.05$, shown that the RGB overlapped block-based algorithm which was optimized using adaptive thresholding gave a slightly lower PSNR value of 44.2726 compared to the original one which had a PSNR of 44.5305, but the capacity obtained is greater than the original algorithm, which is 44,635 bits which has a capacity of 3,371 bits more than the original algorithm.

Tabel 2. Test Results ($T_{bits} = 4, T_{RMSE} = 0.05$)

Image	RGB overlapped block-based PVD		Adaptive Threshold RGB overlapped block-based PVD	
	Capacity (Bits)	PSNR	Capacity (Bits)	PSNR
Ballon	41,264	44.5305	44,635	44.2726
Clock	345,322	36.5155	351,115	36.4214
Apple	58,993	44.4868	65,380	43.5647
Camel	523,296	35.2994	524,527	35.2785

Tabel 3. Capacity Optimization Results

Image	RGB overlapped block-based PVD	Adaptive Threshold RGB overlapped block-based PVD	Optimization
	Capacity (bits)	Capacity (bits)	%
Ballon	41,264	44,635	8.17
Clock	345,322	351,115	1.68
Apple	58,993	65,380	10.83
Camel	523,296	524,527	0.24
		Average Capacity Optimization	5.23

Tabel 4. Degradation Increase

Image	RGB overlapped block-based PVD	Adaptive Threshold RGB overlapped block-based PVD	Degradation
	PSNR	PSNR	%
Ballon	44.5305	44.2726	0.58
Clock	36.5155	36.4214	0.26
Apple	44.4868	43.5647	2.07
Camel	35.2994	35.2785	0.06
		Average Degradation Increase	0.74

The optimization developed in this research using adaptive thresholding is able to provide increased capacity with an average 5.23% of capacity optimization than the original RGB overlapped block-based algorithm. In addition to improving capacity, optimization using adaptive threshold also provides a decrease in quality compared to the original algorithm considering that the capacity used is getting bigger so that the changes to the original pixels will also be even greater than the original algorithm. However, the resulting decrease in quality is not too significant, where the average value of quality loss compared to the original algorithm is only 0.74%.

4. Conclusions

RGB overlapped block-based PVD operates by using pairs of color components from a pixel. This method is very effective in increasing capacity in container images that have low color variance values. Threshold is used to limit the changes that can be applied to the color components which also limits the potential for optimal utilization of the capacity of the color pixels. Optimization using adaptive threshold that was carried out in this study was able to increase the capacity of the container image pixels where the average capacity obtained increased by 5.23%. Using a larger T_{RMSE} value, it is not impossible to obtain an even greater capacity increase. The decrease in degradation that occurs can also be controlled easily using these two thresholds.

Reference

[1] A. Cheddad, J. Condell, K. Curran and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," Signal Processing, vol. 90, no. 3, pp. 727-752, 2010.
[2] A. K. Sahu and M. Sahu, "Digital image steganography and steganalysis: A journey of the past three decades," Open Computer Science, vol. 10, no. 1, pp. 1-47, 2020.
[3] J. Wang, M. Cheng, P. Wu and B. Chen, "A survey on digital image steganography," Journal of Information Hiding and Privacy Protection, vol. 1, no. 2, pp. 87-93, 2019.
[4] W. Bender, D. Gruhl, N. Morimoto and A. Lu, "Techniques for data hiding," IBM Systems Journal, vol. 35, no. 3, pp. 313-336, 1996.
[5] W. N. Lie and L. C. Chang, "Data hiding in images with adaptive numbers of least significant bits based on the human visual system," in IEEE Int. Conf. Image Processing, Rohtak, 1999.
[6] H. B. Kekre, A. A. Athawale and U. A. Athawale, "Increased cover capacity using advanced multiple LSB algorithms," International Conference & Workshop on Emerging Trends in Technology, pp. 25-31, 2011.
[7] J. R. C. Tavares and F. M. B. Junior, "Word-Hunt: a LSB steganography method with low expected

number of modifications per pixel," IEEE Latin America Transactions, vol. 14, no. 2, pp. 1058-1064, 2016.

[8] S. Rustad, D. R. I. M. Setiadi, A. Syukur and P. N. Andono, "Inverted LSB image steganography using adaptive pattern to improve imperceptibility," Journal of King Saud University - Computer and Information Sciences, 2021.

[9] M. R. Islam, T. R. Tanni, S. Parvin, M. J. Sultana and A. Siddiqa, "A modified LSB image steganography method using filtering algorithm and stream of password," Information Security Journal: A Global Perspective, vol. 29, no. 6, pp. 1-12, 2020.

[10] D. C. Wu and W. H. Tsai, "A steganographic method for images by pixel-value differencing," Pattern Recognition Letters, vol. 24, no. 9-10, p. 1613–1626, 2003.

[11] N. M. (Ganguly), G. Paul, S. K. Saha and D. Burman, "A PVD based high capacity steganography algorithm with embedding in non-sequential position," Multimedia Tools and Applications, vol. 79, no. 1, p. 13449–13479, 2020.

[12] A. Pradhan, K. R. Sekhar and G. Swain, "Adaptive PVD Steganography Using Horizontal, Vertical, and Diagonal Edges in Six-Pixel Blocks," Security and Communication Networks, vol. 2017, no. 37, pp. 1-13, 2017.

[13] O. Hosam and N. Ben Halima, "Adaptive block-based pixel value differencing steganography," Security and Communication Networks, vol. 9, pp. 5036-5050, 2016.

[14] A. K. Sahu and G. Swain, "Pixel Overlapping Image Steganography Using PVD and Modulus Function," 3D Research, vol. 9, no. 3, pp. 1-14, 2018.

[15] S. Prasad and A. K. Pal, "An RGB colour image steganography scheme using overlapping block-based pixel-value differencing," Royal Society Open Science, vol. 4, no. 4, pp. 1-14, 2017.